ROYAUME DU MAROC
--*-*-*
HAUT COMMISSARIAT AU PLAN
--*-*-*-*-*-*
INSTITUT NATIONAL
DE STATISTIQUE ET D'ECONOMIE APPLIQUEE
INSEA

Projet de Fin d'Études

*****

Conception et développement d'une plateforme de collaboration intelligente « DXC Synergy »

Préparé par :
- *M. KHOUNA Younes*

Sous la direction de :
- *Mme. HILAL Imane (INSEA)*
- *Mme. IBRAHIMI Safae (DXC TECHNOLOGY)*
- *M. TIOUGA Youssef (DXC TECHNOLOGY)*

Soutenu publiquement comme exigence partielle en vue de l'obtention du

Diplôme d'Ingénieur d'Etat

Filière : Data and Software Engineering (DSE)

Devant le jury composé de :

- *M. ERRAJI Zakarya (INSEA)*
- *Mme HILAL Imane (INSEA)*
- *Mme IBRAHIMI Safae (DXC TECHNOLOGY)*

Juin 2024 / PFE N° 54

Dédicace

Je dédie, ce modeste travail comme preuve de respect et de reconnaissance

À MES CHERS ET AIMABLES PARENTS :

Pour les efforts qu'ils ont consentis pour mon éducation et ma formation, pour leur précieux soutien moral et matériel, pour leurs encouragements continus, et pour leurs sacrifices tout au long de ma vie, que je serai tellement très reconnaissant.

À MES SŒURS

Pour être toujours présentes à mes côtés, pour leur soutien indéfectible et leurs encouragements constants qui m'ont permis de surmonter chaque épreuve. Leur amour et leur présence m'ont donné la force et la motivation nécessaires pour avancer

À MES PROFFESSEURS ET MES ENCADRANTS

Qui n'ont épargné aucun effort pour me soutenir et m'orienter tout au long de mon cursus et de mon stage. Ils ont contribué à la réussite de mon parcours par leurs conseils et leurs instructions précieuses. Leur dévouement et leur engagement ont créé un environnement propice à l'apprentissage et au développement.

À MES AMIS

Qui ont partagé avec moi une période d'étude inoubliable.

ET À VOUS CHERS LECTEURS

Remerciement

Au nom d'Allah le tout miséricordieux,

Tout d'abord, je tiens à exprimer ma profonde gratitude envers Monsieur **KETTANI Mehdi**, Directeur Général de DXC Technology Maroc de m'avoir donné l'opportunité d'effectuer mon stage de projet de fin d'études au sein de cet organisme. Sa confiance et son soutien ont été essentiels pour cette expérience enrichissante.

Je tiens également à remercier chaleureusement Madame **IBRAHIMI Safae** et Monsieur **TIOUGA Youssef**, mes encadrants externes, pour leur soutien continu et leurs conseils avisés. Leur présence quotidienne et leur expertise ont été des atouts majeurs dans la réalisation de ce projet.

Ensuite, je souhaite exprimer ma gratitude envers Madame **HILAL Imane**, mon encadrante interne, pour son encadrement exceptionnel tout au long de ce projet. Sa présence bienveillante, ses conseils éclairés et ses encouragements constants ont été d'une importance capitale pour la réussite du projet.

Je n'oublie pas à remercier chaleureusement le membre du jury et mon examinateur Monsieur **ERRAJI Zakarya** d'avoir accepté d'évaluer mon travail.

Enfin, je voudrais également remercier **toutes les personnes** qui ont contribué, de près ou de loin, à la réussite de ce projet. Leur collaboration, leur soutien et leurs idées ont été essentiels pour surmonter les défis rencontrés.

Résumé

Ce présent rapport constitue le résultat de mon stage de fin d'étude de cinq mois au sein de l'entreprise DXC Technology Maroc. Ce stage s'inscrivait dans le cadre de la validation de ma troisième année du cycle d'ingénieur d'État à l'Institut National de Statistique et d'Économie Appliquée (INSEA).

L'objectif principal de ce projet, intitulé « DXC SYNERGY », est de mettre en place une plateforme de collaboration intelligente spécialement conçue pour les collaborateurs de l'organisation. Cette plateforme vise à garantir une collaboration optimale et à maximiser la productivité des équipes de chaque département, assurant ainsi **l'alignement stratégique**, **opérationnel et fonctionnel** entre les différents départements de l'organisation.

« DXC SYNERGY » cherche à digitaliser divers processus métiers, tels que la gestion de projet, le traitement des demandes de congé, et la gestion documentaire, afin de réduire les tâches administratives manuelles et d'améliorer l'efficacité globale. En favorisant une meilleure communication et une coordination plus fluide, cette initiative soutient également l'innovation continue et l'amélioration des processus internes, renforçant la compétitivité et la réactivité de l'entreprise sur le marché.

Mots clés : DXC Technology, DXC Synergy, Collaboration, microservices, SCRUM, chatbot LLM.

.

Abstract

This report is the result of my five-month graduation internship at DXC Technology Morocco. This internship was part of the validation of my third year in the State Engineer program at the National Institute of Statistics and Applied Economics (INSEA).

The main objective of this project, titled "DXC SYNERGY," is to implement an intelligent collaboration platform specifically designed for the organization's employees. This platform aims to ensure optimal collaboration and maximize the productivity of teams in each department, thereby ensuring strategic, operational, and functional alignment across the organization's various departments.

« DXC SYNERGY » seeks to digitize various business processes, such as project management, leave request processing, and document management, to reduce manual administrative tasks and improve overall efficiency. By fostering better communication and smoother coordination, this initiative also supports continuous innovation and the improvement of internal processes, enhancing the company's competitiveness and responsiveness in the market.

Keywords : DXC Technology, DXC Synergy, Collaboration, microservices, SCRUM, chatbot LLM.

ملخص

يُعد هذا التقرير نتيجة لفترة تدريبي النهائي لمدة خمسة أشهر في شركة DXC Technology المغرب. كان هذا التدريب جزءًا من متطلبات السنة الثالثة في برنامج المهندس الحكومي في المعهد الوطني للإحصاء والاقتصاد التطبيقي(INSEA) .

الهدف الرئيسي من هذا المشروع، الذي يحمل عنوان"DXC SYNERGY" ، هو تطوير منصة ذكية للتعاون مصممة خصيصًا لموظفي المنظمة. تهدف هذه المنصة إلى ضمان تعاون مثالي وزيادة إنتاجية الفرق في كل قسم، مما يضمن التوافق الاستراتيجي والتشغيلي والوظيفي بين مختلف أقسام المنظمة.

يسعى "DXC SYNERGY" إلى رقمنة مختلف العمليات الداخلية للشركة، مثل إدارة المشاريع، ومعالجة طلبات الإجازة، وإدارة الوثائق، من أجل تقليل المهام الإدارية اليدوية وتحسين الكفاءة العامة. من خلال تعزيز التواصل الأفضل والتنسيق السلس، تدعم هذه المبادرة أيضًا الابتكار المستمر وتحسين العمليات الداخلية، مما يعزز من تنافسية الشركة واستجابتها في السوق.

الكلمات المفتاحية: DXC Technology, DXC Synergy, تعاون SCRUM, microservices, روبوت الدردشة LLM,

Table de matière

Dédicace .....
Remerciement .....
Résumé .....
Abstract .....
ملخص .....
Table de matière .....
Liste des figures .....
Liste des tableaux .....
Liste des acronymes .....
Introduction 1
Chapitre I : Contexte général et organisme d'accueil 2
I. Organisme d'accueil 3
I.1. Présentation de DXC Technology 3
I.2. DXC Technology Maroc en 360° 4
I.3. Secteur d'activité 5
I.4. Structure 6
I.5. Clients et partenaires stratégiques 7
I.5.A Clients 7
I.5.B Partenaires stratégiques 8
II. Description du projet 9
II.1 Contexte général 9
II.2 Problématique 10
II.3 Objectifs du projet : DXC Synergy 10
III. Méthodologie adoptée et planification du projet 11
III.1 Méthodologie de développement 11
III.1.A Motivation du choix 12
III.1.B Répartition des rôles Scrum dans le projet 12
III.2 Planification du projet 13
III.2.A Planification des sprints 13
III.2.B Durée totale du projet 14
III.2.C Outil de planification de projet : GANTT Project 14
III.3 Outil de gestion de projet : JIRA 15
IV. Synthèse 17

Chapitre II : Spécification fonctionnelle 18
I. Périmètre du projet 19
II. Besoins fonctionnels 19
II.1 Module gestion de l'identité 20
II.2 Module gestion du patrimoine documentaire 21
II.3 Module gestion des projets 21
II.4 Module traitement des demandes de congé 22
II.5 Module feeling-board 22
II.6 Module forum de partage 23
II.7 Module chat room 23
II.8 Module messagerie 23
II.9 Module chatbot RAG 24
II.10 Module notification 24
III. Besoins non fonctionnels 24
III.1 Point de vue utilisateur 25
III.2 Point de vue développeur 26
III.3 Point de vue communication 26
IV. Maquettes 27
IV.1 Maquette de la page d'authentification 27
IV.2 Maquette de la page d'ajout d'un nouveau collaborateur 28
IV.3 Maquette de la page de gestion des documents 29
IV.4 Maquette de la page de progression des tâches (vue chef d'équipe) 30
IV.5 Maquette de la page de progression des tâches (vue collaborateur) 31
V. Synthèse 32
Chapitre III : Conception (vue fonctionnelle) 33
I. Motivation du choix du langage de conception 34
II. Diagramme de cas d'utilisation 34
II.1 Module Gestion de l'identité 35
II.2 Module Gestion du patrimoine documentaire 37
II.3 Module Gestion des projets 39
II.4 Module traitement des demandes de congé. 41
II.5 Module Feeling board 42
II.6 Module forum de partage 44
II.7 Module chat room 45
II.8 Module messagerie et chat room 46
II.9 Module chatbot RAG 47
II.10 Module notification 49

III. Synthèse **50**
Chapitre IV : Conception (vue structurelle) 51
I. Motivation du choix des diagrammes **52**
II. Diagramme de classes et son instance **52**
II.1 Module Gestion de l'identité. 52
II.2 Module Gestion du patrimoine documentaire 54
II.3 Module Gestion des projets 55
II.4 Module traitement des demandes de congé 57
II.5 Module feeling board 58
II.6 Module forum de partage 59
II.7 Module chat room 61
II.8 Module messagerie 62
II.9 Module chatbot RAG 62
II.10 Module notification 62
III. Diagramme de paquetages **63**
III.1 Diagramme de paquetage global 63
III.2 Etude de cas : Diagramme de paquetage de « identity-service>> 63
IV. Synthèse **65**
Chapitre V : Conception (vue dynamique) 66
I. Motivation du choix des diagrammes **67**
II. Diagrammes de séquences **67**
II.1 Diagramme de séquences du cas d'utilisation : S'authentifier 67
II.2 Diagramme de séquences du cas d'utilisation : Ajouter une équipe 68
II.3 Diagramme de séquences du cas d'utilisation : Consulter un document 70
II.4 Diagramme de séquences du cas d'utilisation : Ajouter un projet 71
II.5 Diagramme de séquences du cas d'utilisation : Créer product-backlog 72
II.6 Diagramme de séquences du cas d'utilisation : Ajouter fonctionnalité 73
II.7 Diagramme de séquences du cas d'utilisation : planifier sprint 74
II.8 Diagramme de séquences du cas d'utilisation : Terminer tâche 76
II.9 Diagramme de séquences du cas d'utilisation : Envoyer une demande de congé 76
II.10 Diagramme de séquences du cas d'utilisation : Traiter une demande de congé 78
III. Diagramme d'état-transition **79**
III.1 Diagramme de d'état-transition de la classe : Collaborateur 79
III.2 Diagramme de d'état-transition de la classe : Projet 80
III.3 Diagramme de d'état-transition de la classe : Demande de congé 81
III.4 Diagramme de d'état-transition de la classe : Message 82
IV. Synthèse **83**

Chapitre VI : Environnement technique et architecture logicielle 84
I. Environnement technique 85
I.1 Outils de développement 85
I.1.A Intellij idea 85
I.1.B Visual studio code 85
I.1.C Postman 86
I.1.D Docker 86
I.1.E Apache kafka 87
I.2 Langages de programmation 87
I.2.A Java 88
I.2.B Javascript 88
I.2.C Typescript 89
I.2.D Html & Css 89
I.3 Frameworks de développement 90
I.3.A Angular 90
I.3.B Spring boot 91
I.3.C Spring data jpa 91
I.3.D Spring Security 92
I.3.E Spring Cloud 92
I.3.F Spring AI 93
I.4 Plugins et dépendances 93
I.4.A Lombok 94
I.4.B Swagger 94
I.4.C Zipkin 95
I.4.D Prometheus & Grafana 95
I.4.E Websockets 96
I.5. Systèmes de gestion de base de données (SGBD) 96
I.5.A PostgreSQL 96
I.5.B MongoDB 97
II. Architecture de l'application 97
II.1 Motivations du choix de l'architecture 98
II.2 Pattern de décomposition : Business Capability 99
II.3 Architecture globale de l'application 100
II.4 Communication inter-services EDA (Event Driven Architecture) 101
II.4.A Etude de cas : Communication entre « identity-service » et « notification-service » 101
II.6 Architecture du chatbot RAG 102

III. Synthèse 104
Chapitre VII : Réalisation 105
I. Point de vue utilisateur 106
I.1 Scénario : Nouveau projet affecté 106
I.2 Scénario : Ajout d'un nouveau collaborateur 111
I.3 Scénario : Traitement d'une demande de congé 115
I.4 Scénario : Demander l'aide au Reference chatbot 118
I.5 Scénario : Changer sa photo de profil dans le forum 119
I.6 Scénario : Envoyer un message privé 120
I.7 Scénario : Publier son sentiment dans le feeling board 121
I.8 Scénario : Télécharger un document vue manager 123
II. Point de vue développeur 124
II.1 Codage des microservices 124
II.2 Externalisation de la configuration 124
II.3 Inscription des microservices et équilibrage de charge. 125
II.4 Documentation 126
II.5 Implémentation du proxy-service 126
II.6 Traçabilité des requêtes client http 127
II.7 Sécurité et confidentialité des données. 127
II.8 Configuration du kafka Listener 127
II.9 Implémentation du chatbot RAG 128
III. Synthèse 128
Conclusion 129
I. Bilan 130
II. Perspectives 130
Glossaire 131
Bibliographie 133

Liste des figures

Figure 1: Historique de DXC Technology 3
Figure 2: Secteur d'activité de DXC Technology Maroc 5
Figure 3: Organigramme DXC Technology Maroc 7
Figure 4: Principaux clients de DXC Technology Maroc 8
Figure 5: Principaux partenaires de DXC Technology 8
Figure 6: Approche Top-Down 9
Figure 7: Diagramme d'Ishikawa 11
Figure 8: Architecture du Framework Scrum [8] 12
Figure 9: Diagramme de Gantt 15
Figure 10: Aperçu du sprint (4) avec JIRA 16
Figure 11: Modules identifiés dans notre projet 20
Figure 12: Besoins non fonctionnels **[9]** 24
Figure 13: Maquette de la page d'authentification 28
Figure 14: Maquette de la page d'ajout d'un collaborateur 28
Figure 15: Maquette de la page de gestion des documents 29
Figure 16: Maquette de la page de progression des tâches vue chef d'équipe 30
Figure 17: Maquette de la page de progression des tâches vue collaborateur 31
Figure 18: Différentes Vues UML [12] 34
Figure 19: Diagramme de cas d'utilisation <<identity-service>> 36
Figure 20: Paquetage cas d'utilisation <<Gérer collaborateurs>> 37
Figure 21: Paquetage cas d'utilisation <<Gérer équipes>> 37
Figure 22: Diagramme de cas d'utilisation <<patrimoineDoc-service>> 38
Figure 23: Paquetage cas d'utilisation <<Gérer documents>> 39
Figure 24: Diagramme de cas d'utilisation <<projectScrum-service-service>> 40
Figure 25 : Paquetage cas d'utilisation <<Gérer product-backlog>> 41
Figure 26: Paquetage cas d'utilisation <<Gérer taches>> 41
Figure 27: Diagramme de cas d'utilisation <<hollidays-service>> 42
Figure 28: Diagramme de cas d'utilisation <<feelingBoard-service>> 43
Figure 29: Diagramme de cas d'utilisation <<forum-service>> 45
Figure 30: Diagramme de cas d'utilisation <<chatRoom-service>> 46
Figure 31: Diagramme de cas d'utilisation <<privateChat-service>> 47
Figure 32: Diagramme de cas d'utilisation <<chatbtRAG-service>> 48
Figure 33: Paquetage cas d'utilisation <<Gérer base de connaissance>> 49
Figure 34: Diagramme de cas d'utilisation <<notification-service>> 50
Figure 35: Diagramme de classes <<identity-service>> 53
Figure 36: Diagramme d'objets <<identity-service>> 53
Figure 37: Diagramme de classes <<patrimoineDoc-service>> 54
Figure 38: Diagramme d'objets <<patrimoineDoc-service>> 55
Figure 39: Diagramme de classes <<projectScrum-service>> 56
Figure 40: Diagramme d'objets <<projectScrum-service>> 56
Figure 41: Diagramme de classes <<hollidays-service>> 57
Figure 42: Diagramme d'objets <<hollidays-service>> 58
Figure 43: Diagramme de classes <<feelingBoard-service>> 58
Figure 44: Diagramme d'objets <<feelingBoard-service>> 59
Figure 45: Diagramme de classes « forum-service » 60

Figure 46: Diagramme d'objets <<forum-service>> 60
Figure 47: Diagramme de classes <<chatRoom-service>> 61
Figure 48: Diagramme d'objets <<chatRoom-service>> 61
Figure 49: Diagramme de classes <<notification-service>> 62
Figure 50: Diagramme de paquetages global 63
Figure 51: Diagramme de paquetages "identity-service" 64
Figure 52: Diagramme de paquetages "User" 64
Figure 53: Diagramme de séquences <<S'authentifier>> 68
Figure 54: Diagramme de séquences <<Ajouter une équiper>> 69
Figure 55: Diagramme de séquences <<Consulter document>> 70
Figure 56: Diagramme de séquences <<Ajouter projet>> 72
Figure 57: Diagramme de séquences <<Créer product-backlog>> 73
Figure 58: Diagramme de séquences <<Ajouter fonctionnalité>> 74
Figure 59: Diagramme de séquences <<Planifier sprint>> 75
Figure 60: Diagramme de séquences <<Terminer tâche>> 76
Figure 61: Diagramme de séquences <<Envoyer demande de congé>> 78
Figure 62: Diagramme de séquences <<Traiter demande de congé>> 79
Figure 63: Diagramme d'état-transition de la classe <<Collaborateur> 80
Figure 64: Diagramme d'état-transition de la classe <<Projet> 81
Figure 65: Diagramme d'état-transition de la classe <<Demande de congé>> 82
Figure 66: Diagramme d'état-transition de la classe <<Message>> 82
Figure 67: Choix de l'architecture logicielle [10] 98
Figure 68: Architecture de l'application 100
Figure 69: Architecture Appach kafka [11] 101
Figure 70: Communication asynchrone identity-service et notification-service 102
Figure 71: Architecture du chatbot RAG 103
Figure 72: Page d'authentification 106
Figure 73: Page espace<<Accueil>> vue Manager 1/2 106
Figure 74: Page espace<<Accueil>> vue Manager 2/2 107
Figure 75: Page espace<<Projets>> vue Manager ½ 107
Figure 76: Formulaire d'ajout d'un projet 108
Figure 77: Réception de la notification via SMTP pour Scrum master 108
Figure 78: Réception de la notification dans l'application pour Scrum master 108
Figure 79: Création du product-backlog 109
Figure 80: Ajout des fonctionnalités du projet 109
Figure 81: Notification des équipes ''nouveau projet'' 109
Figure 82: Planification des sprints par le chef d'équipe 110
Figure 83: Suivie des tâches par le chef d'équipe 1/2 110
Figure 84: Suivie des tâches par le chef d'équipe 2/2 111
Figure 85: Page espace "mon département" 1/3 111
Figure 86: Page espace "mon département" 2/3 111
Figure 87: Page espace "mon département" 3/3 112
Figure 88: Espace équipe « DXC Synergy » 1/3 112
Figure 89: Espace équipe « DXC Synergy » 2/3 112
Figure 90: Espace équipe « DXC Synergy » 3/3 113
Figure 91: Formulaire ajout du nouveau collaborateur 114
Figure 92: Ajout du nouveau collaborateur avec succès 114
Figure 93: Notification du chef d'équipe "Nouveau membre" 114
Figure 94: Authentification du nouveau collaborateur 115

Figure 95: Marquer la notification de bienvenue comme lue 115
Figure 96: Formulaire demande de congé 116
Figure 97: demande de congé envoyée avec succès 116
Figure 98: notification du remplaçant, manager et demandeur via SMTP 116
Figure 99: Progression de la demande de congé 116
Figure 100: Réception et traitement de la demande de congé 117
Figure 101: notification du traitement de la demande de congé 117
Figure 102: Mise à jour de la progression de la demande de congé 117
Figure 103: Ouvrir la bulle de conversation chatbot 118
Figure 104: poser une question au Reference chatbot 118
Figure 105: Forum de partage 1/2 119
Figure 106: Forum de partage 2/2 119
Figure 107: Changement de la photo de profil 1/2 120
Figure 108: Changement de la photo de profil 2/2 120
Figure 109: Choisir une conversation 120
Figure 110: Envoyer un nouveau message 121
Figure 111: Feeling board des sentiments initiaux 121
Figure 112: publier son sentiment dans le feeling board1/2 122
Figure 113: publier son sentiment dans le feeling board 2/2 122
Figure 114 : Mise à jour du feeling board (mode sombre) 122
Figure 115: Liste des documents publiés dans l'espace document 123
Figure 116: Téléchargement du document 123
Figure 117: Structure du code 124
Figure 118: Externalisation de la configuration des microservices 125
Figure 119: Equilibrage de charge et résilience grâce à Eureka 125
Figure 120: Documentation de <<identity-service>> 126
Figure 121: Routage des microservice 126
Figure 122: Traçabilité des requêtes HTTP 127
Figure 123: Sécurité des données avec l'algorithme Bcrypt 127
Figure 124: Configuration du kafka Listener 128
Figure 125: Implémentation du chatbot RAG 128

Liste des tableaux

Tableau 2: Fiche signalétique de DXC Technology Maroc 4
Tableau 3: Répartition des rôles Scrum dans le projet 13
Tableau 4: Planification des sprints 14
Tableau 5: Acteurs du système 19
Tableau 6: Description des cas d'utilisation de <<Identity-service>> 36
Tableau 7: Description des cas d'utilisation de <<patrimoineDoc-service>> 38
Tableau 8: Description des cas d'utilisation de <<projectScrum-service>> 40
Tableau 9: Description des cas d'utilisation de <<hollidays-service>> 42
Tableau 10: Description des cas d'utilisation de <<feelingBoard-service>> 43
Tableau 11: Description des cas d'utilisation de <<forum-service>> 44
Tableau 12: Description des cas d'utilisation de <<chatRoom-service>> 46
Tableau 13: Description des cas d'utilisation de <<privateChat-service>> 47
Tableau 14: Description des cas d'utilisation de <<chatbotRAG-service>> 48
Tableau 15: Description des cas d'utilisation de <<notification-service>> 49
Tableau 16: Description des scénarios "S'authentifier" 68
Tableau 17: Description des scénarios "Ajouter une équiper" 69
Tableau 18: Description des scénarios "Consulter un document" 70
Tableau 19: Description des scénarios "Ajouter un projet" 71
Tableau 20: Description des scénarios "Créer product-backlog" 72
Tableau 21: Description des scénarios "Ajouter fonctionnalité" 74
Tableau 22: Description des scénarios "Planifier sprint" 75
Tableau 23: Description des scénarios "Terminer tâche" 76
Tableau 24: Description des scénarios "Envoyer une demande de congé" 77
Tableau 25: Description des scénarios "Traiter une demande de congé" 78
Tableau 26: Transitions d'état de la classe « Collaborateur » 80
Tableau 27: Transitions d'état de la classe « Projet » 81
Tableau 28: Transitions d'état de la classe « Demande de congé » 81
Tableau 29: Transitions d'état de la classe « Message » 82
Tableau 30: Décomposition des microservices 100

Liste des acronymes

Acronyme	Désignation
API	Application Programming Interface
AWS	Amazon Web Services
BI	Business Intelligence
CDG	Caisse de Dépôt et de Gestion
DTO	Data Transfer Objects
HTTP	Hypertext Transfer Protocol
IA	Intelligence Artificielle
IDE	Integrated Development Environment
IT	Information Technologies
JWT	Json Web Token
LLM	Large Language Model
RAG	Retrieval-Augmented Generation
REST	REpresentational State Transfer
SGBDR	Système de Gestion de Base de Données Relationnelle
SQL	Structured Query Language
SMTP	Simple Mail Transfer Protocol
UI	User Interfaces
UML	Unified Modeling Language
XSS	Cross-Site Scripting

Tableau 1: Liste des abréviations

Introduction

Dans un contexte de marché compétitif, renforcer sa position en tant que leader de son secteur nécessite de mettre l'accent sur la qualité des produits livrés aux clients. En conséquence, garantir un produit de haute qualité dans un Time-To-Market minimum nécessite une collaboration étroite entre les différents acteurs de l'entreprise, allant de l'étude de faisabilité à la production.

Ainsi le projet **« DXC Synergy »** s'inscrit dans ce contexte, visant à faciliter la collaboration, la coordination et le partage de l'informations entre les différents départements, optimisant ainsi la productivité et permettant une gestion plus agile des processus de management, métier et support de l'entreprise.

Après la rédaction de notre cahier des charges, Nous avons rigoureusement établi la phase de spécification fonctionnelle. Elle a débuté par l'identification des acteurs impliqués, suivie de la description des besoins fonctionnels et non fonctionnels de chaque processus interne à digitaliser. Ensuite, l'élaboration des maquettes qui a nécessité l'implication du client pour les valider. Ces maquettes représentent un modèle de base dont nous nous somme basés dans la phase de développement (coté frontend).

Pour l'étape de conception, nous avons élaboré les diagrammes UML nécessaires pour chaque perspective, incluant la vue fonctionnelle, structurelle et dynamique. Cela nous a permis d'obtenir une vision transversale du projet. Plus spécifiquement, la vue fonctionnelle a été représentée par le diagramme de cas d'utilisation, la vue structurelle par le diagramme de classes, d'objets et de paquetages, et enfin la vue dynamique par le diagramme de séquences et de transition d'états.

Ensuite, la phase de développement, nous avons utilisé l'écosystème Java pour le développement du backend de notre application, tandis que le frontend a été réalisé en utilisant JavaScript. Cette combinaison judicieuse nous a grandement simplifié cette étape du projet.

Enfin, la phase de test et de déploiement, nous avons planifier la réalisation des tests exhaustifs, pour chaque module du projet de manière indépendante, pour les deux mois prochains, garantissant ainsi son évolutivité et son autonomie. De même pour le déploiement, nous visons opter pour l'approche de conteneurisation avec Docker, permettant le déploiement autonome de chaque module.

Ainsi, nous avons structuré notre rapport du projet en cinq chapitres à savoir :

➢ Le chapitre I décrit l'organisme d'accueil et le contexte général du projet, sa présentation, son objectif et sa conduite tout au long du processus de développement.

➢ Le chapitre II détaille la phase de spécification fonctionnelle en délimitant le périmètre, les besoins fonctionnels et non fonctionnels ainsi que les maquettes du projet.

➢ Le chapitre III détaille la phase de conception en illustrant tous les diagrammes UML nécessaires.

➢ Le chapitre IV définit l'environnement et les outils de la solution. Il présente les outils et les technologies utilisées pour implémenter la solution Ainsi que l'architecture logicielle détaillée.

➢ Le chapitre IV concerne la réalisation du projet selon deux points de vue, celui de l'utilisateur en termes de UI (interfaces utilisateurs) et celui du développeur en termes de design patterns assurés lors de la phase de développement.

➢ La conclusion constitue un bilan du travail effectué et offre un aperçu des perspectives.

Chapitre I : Contexte général et organisme d'accueil

Dans ce chapitre, nous débutons par une présentation de l'organisme d'accueil, incluant son identité, son secteur d'activité et sa structure. Nous poursuivons ensuite avec une description du contexte de notre projet, en exposant sa problématique et ses objectifs.

I. Organisme d'accueil

Cette phase est essentielle pour acquérir une compréhension approfondie de l'entreprise d'accueil, découvrir son identité, son secteur d'activité, ainsi que ses atouts sur le marché.

I.1. Présentation de DXC Technology

DXC Technology est une société de service de technologie américaine issue de la fusion de CSC (Computer Sciences Corporation) et de Hewlett-Packard Enterprise (HPE) qui est spécialisé dans les métiers du conseil, tandis qu'HP provient des métiers de l'infogérance et de l'infrastructure. **[1]**

En tant que leader mondial des services informatiques, DXC Technology se positionne comme un acteur clé dans le domaine de la transformation numérique pour les entreprises et les gouvernements.

L'entreprise compte 137 000 salariés et 6 000 clients, répartis dans plus de 70 pays, Le réseau de partenaires de DXC permet de favoriser la collaboration et de tirer parti de l'indépendance technologique. La société a établi plus de 200 relations avec le réseau mondial de partenaires DXC, dont 14 partenaires stratégiques : Amazon Web Services, ATT, Dell Technologies, Google Cloud, HP, HPE, IBM, Micro-Focus, Microsoft, Oracle, PwC, SAP, ServiceNow et VMware

La figure (1) illustre l'historique de l'entreprise DXC Technology.

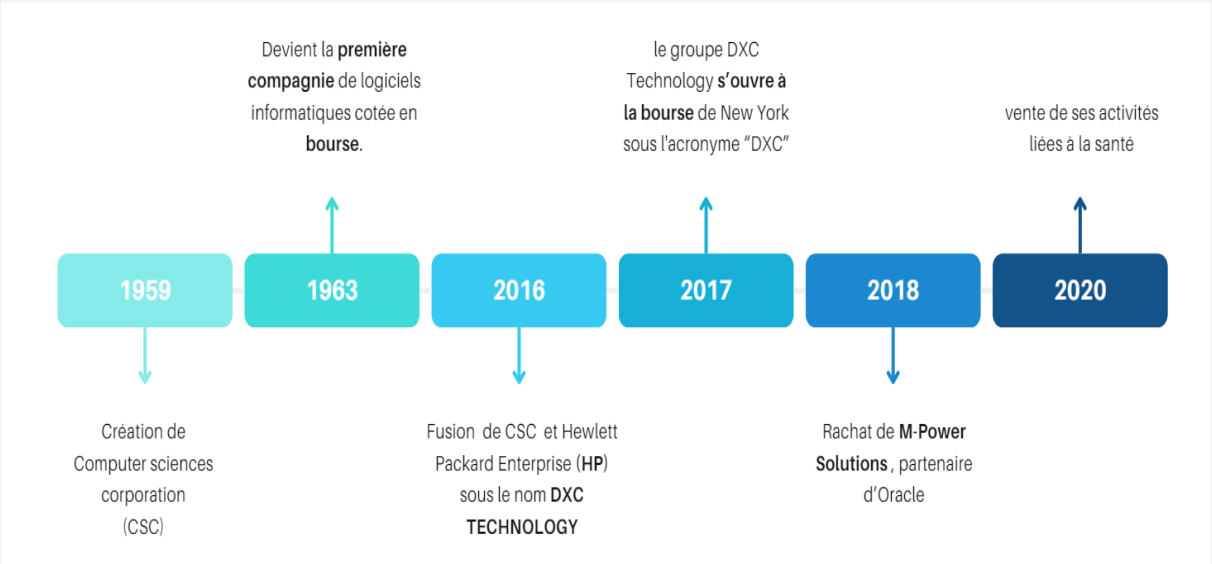


Figure 1: Historique de DXC Technology

I.2. DXC Technology Maroc en 360°

DXC Technology Maroc est une entreprise de technologie dont le siège se trouve à Technopolis Sala al-jadida. Il s'agit d'une coentreprise du groupe DXC Technology et de la Caisse de dépôt et de gestion (CDG) du Maroc. DXC Technology au Maroc, implantée depuis 2007, est une joint-venture entre DXC Technology, leader mondial des services d'intégration d'infogérance et le Groupe CDG. **[2]**

Avec environ 1300 collaborateurs répartis sur 3 sites de production (Technopolis, Hay Riad Casa, NearShore), DXC Technology au Maroc a pour vocation d'accompagne les grands comptes et donneurs d'ordre publics et privés dans leur transformation digitale et d'adresser ainsi l'ensemble des enjeux IT que sont l'Hébergement, l'Infogérance, la Modernisation Applicative, Business Intelligence Analytics, Workplace Mobilty Services, Business Process services, la cyber sécurité et conseils. **[3]**

Le tableau (2) montre le statut juridique de l'entreprise DXC Technology Morocco. **[4]**

Création	1er Juillet 2007 (ES CDG)
Fondateurs	DXC Technology / CDG
Forme juridique	Joint-Venture
Position hiérarchique	Filiale
Tête de groupe mondiale	DXC Technology Company
Directeur	M. KETTANI Mehdi
Siège social	Bâtiment B9, Technopolis 11 100, Sala Al Jadida Maroc.
Tel	+212 5 30 57 64 00
Actionnaires	DXC Technology (51%) CDG (49%)
Domaine d'activité	Programmation, conseil et autres activités informatiques.
Effectif	+ 1200 personnes
Chiffre d'affaires	+ 300 millions de dirhams

Tableau 2: Fiche signalétique de DXC Technology Maroc

I.3. Secteur d'activité

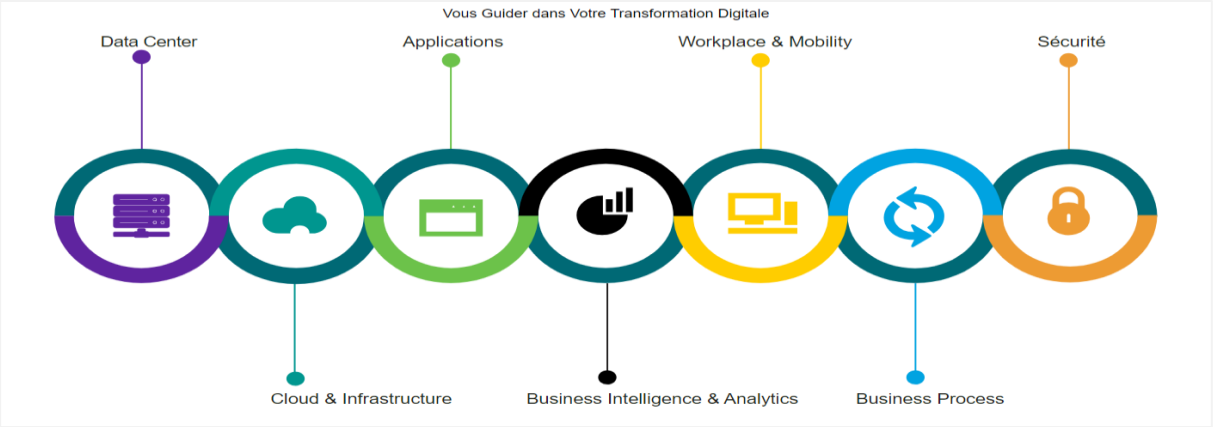


Figure 2: Secteur d'activité de DXC Technology Maroc

DXC Technology Morocco opère dans un secteur d'activité varié et complet, offrant une gamme étendue de services pour ses clients, qu'ils soient dans le secteur public ou privé.

Parmi ses sévices on trouve :

➢ **Data center :** l'entreprise gère des centres de données (Data Centers) de pointe pour héberger et protéger les informations critiques des entreprises.

➢ **Cloud & Infrastracture :** elle excelle également dans les solutions de cloud et d'infrastructures, permettant aux clients de migrer, gérer et optimiser leurs environnements cloud.

➢ **Applications :** DXC propose des services de développement, de gestion et de modernisation des applications pour répondre aux besoins évolutifs des entreprises.

➢ **Business intelligence & Analytics :** le BI constitue un autre domaine clé, fournissant des analyses approfondies et des données exploitables pour une prise de décision éclairée.

➢ **Workplace & Mobility :** DXC améliore l'efficacité opérationnelle grâce à ses services de Workplace & Mobility, qui offrent des solutions de travail flexibles et mobiles.

➢ **Business process :** DXC optimise les processus métier (Business Process) en offrant des solutions complètes qui visent à améliorer l'efficacité opérationnelle et à réduire les coûts pour ses clients.

➢ **Sécurité :** offre des services complets pour protéger les informations et les infrastructures de ses clients contre les menaces cybernétiques. Les solutions de sécurité de DXC comprennent la gestion des identités et des accès, la sécurité des réseaux, la protection des données et la réponse aux incidents.

I.4. Structure

La structure de DXC Technology Maroc est principalement divisée en trois (3) niveaux à savoir :

➢ **Niveau management** *(Vue stratégie) :* Au sommet, le Directeur Général, Monsieur KETTANI Mehdi, supervise les départements juridique (LEGAL) et Audit et Risques (AUDIT & RISK). Ces départements jouent des rôles essentiels dans le maintien de la conformité aux lois et régulations, ainsi que dans la gestion des risques et la prévention des fraudes, garantissant ainsi la stabilité et la sécurité des opérations.

➢ **Niveau opérationnel** *(Vue Métier)* : les départements Project Run & Lead, Manage Lead et Business Operations sont responsables de la gestion efficace des projets et des opérations quotidiennes. Chaque département est subdivisé en équipes dirigées par un chef d'équipe unique et supervisées par plusieurs managers, assurant ainsi une coordination optimale des ressources et une supervision étroite pour atteindre les objectifs de l'entreprise et satisfaire les clients.

➢ **Niveau support** *(Vue fonctionnelle)* **:** e niveau fonctionnel soutient l'ensemble de l'organisation avec les départements des Ressources Humaines (RH) et des Finances. Les RH sont divisées en cinq équipes spécialisées, tandis que les Finances comprennent les équipes Accounting, Controlling et Procurement, assurant une gestion complète des besoins du personnel et des opérations financières de l'entreprise.

La figure (3) montre l'organigramme de l'entreprise DXC Technology Maroc.

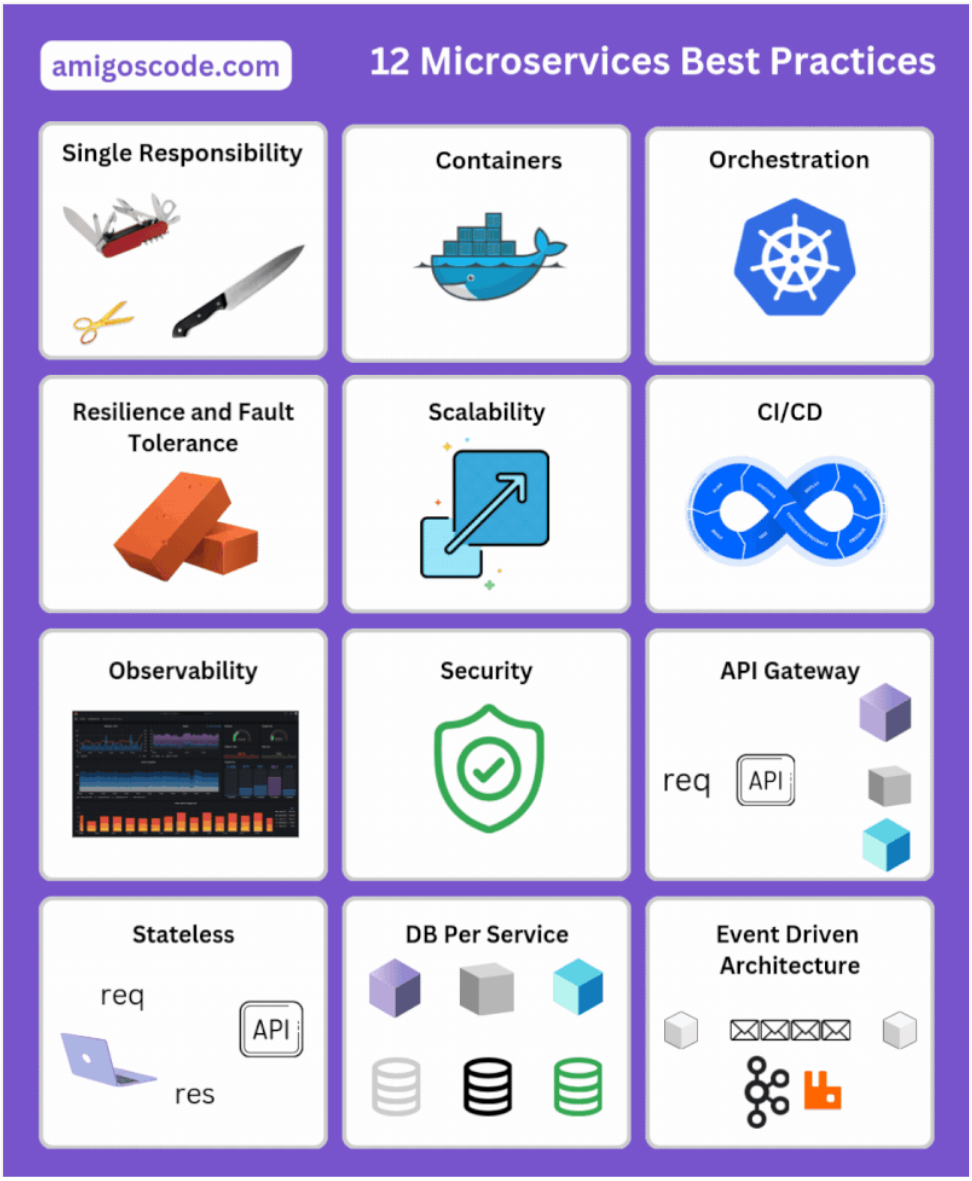


Figure 3: Organigramme DXC Technology Maroc

I.5. Clients et partenaires stratégiques

Dans cette phase, nous mettrons en avant les atouts de DXC Technology, notamment ses clients et partenaires, qui en font une multinationale de grande renommée.

I.5.A Clients

DXC Technology Maroc sert un portefeuille prestigieux de clients, parmi lesquels figurent des institutions et entreprises majeures telles que la Caisse de Dépôt et de Gestion (CDG), Royal Air Maroc (RAM), l'Office Chérifien des Phosphates (OCP), l'Office National des Chemins de Fer (ONCF) et le Ministère de l'Économie et des Finances (MEF). **[5]**

La figure (4) montre un aperçu des principaux clients de DXC Technology Maroc.

Figure 4: Principaux clients de DXC Technology Maroc

I.5.B Partenaires stratégiques

DXC Technology Maroc collabore étroitement avec plusieurs partenaires stratégiques de renom, notamment HP, ORACLE, AWS, DELL Technologies Google Cloud, Salesforce et IBM. Ces partenariats jouent un rôle crucial dans le renforcement de l'offre de services de DXC Technology Maroc.

En s'associant avec ces leaders mondiaux, DXC Technology Maroc est en mesure de fournir des solutions innovantes, fiables et à la pointe de la technologie à ses clients. Cette collaboration stratégique permet à l'entreprise de répondre efficacement aux besoins diversifiés de ses clients, de maintenir une position compétitive sur le marché et d'assurer une qualité de service optimale. **[6]**

La figure (5) illustre les principaux partenaires de DXC Technology Maroc.

Figure 5: Principaux partenaires de DXC Technology

II. Description du projet

Cette étape permet de fournir une vision claire de son origine, de sa raison d'être et de la problématique à laquelle il répond.

II.1 Contexte général

Après analyse approfondie, les décideurs de DXC Technology Maroc ont formulé **une vision stratégique** visant à améliorer le **chiffre d'affaires** de l'entreprise et à renforcer sa position sur le marché. Ils ont identifié que **l'optimisation de la collaboration** entre les différents départements et équipes de l'organisation jouerait un rôle crucial dans l'amélioration de la rentabilité.

En conséquence, ils ont conclu qu'un outil fiable et efficace serait essentiel pour faciliter cette initiative. Cette approche stratégique vise à améliorer la communication, la coordination et la productivité au sein de l'entreprise, assurant une meilleure intégration des processus et une utilisation optimale des ressources, tout en soutenant les objectifs de croissance et de performance de DXC Technology Maroc.

Il s'agit donc d'une démarche **Top-down**, initiée par la couche stratégique (dirigeants, décideurs) et visant à être implémentée jusqu'à la couche technique (logiciels et technologies).

La figure (6) illustre cette approche. **[7]**

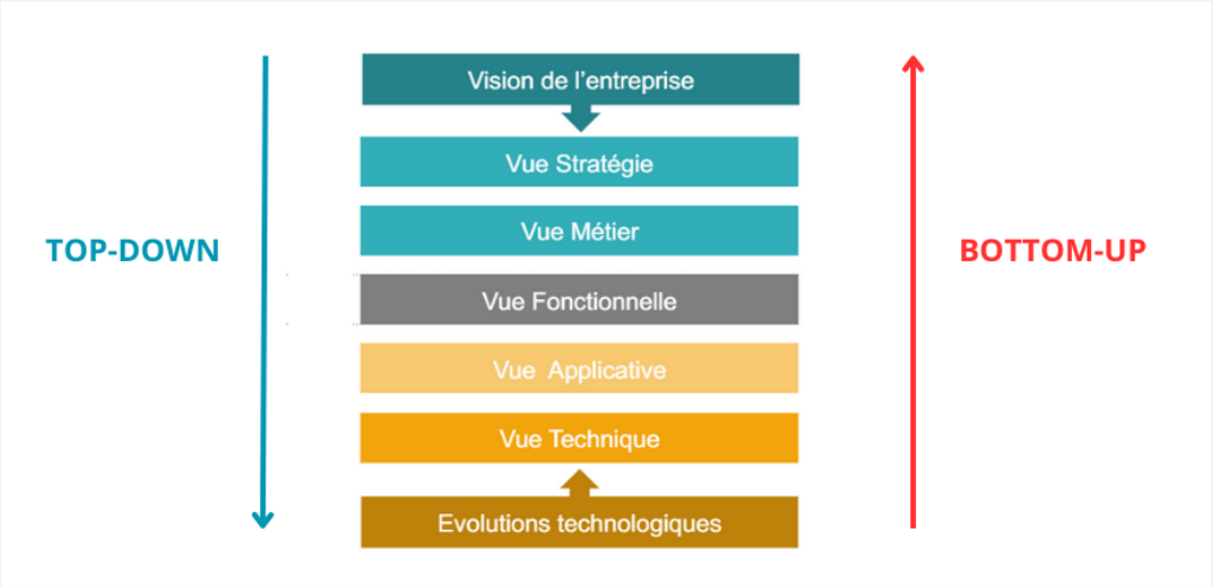


Figure 6: Approche Top-Down

II.2 Problématique

Actuellement, DXC Technology Maroc utilise principalement Microsoft Teams et JIRA pour la communication et la collaboration en ligne, et des interactions en présentiel pour la coordination.

Bien que ces outils soient utiles pour la communication basique, **ils ne suffisent pas** à répondre efficacement aux **besoins actuels de l'entreprise**. Ainsi l'absence d'un outil intégré et centralisé limite la capacité de l'entreprise à suivre et à gérer les processus essentiels de manière fluide et cohérente.

Pour répondre à la vision stratégique des dirigeants de DXC Technology Maroc, il est impératif de développer une solution complète qui **centralise** et optimise la gestion de ces fonctions clés.

II.3 Objectifs du projet : DXC Synergy

Pour répondre à la vision stratégique des dirigeants de DXC Technology Maroc, nous avons conçu le projet « DXC Synergy » visant à digitaliser et optimiser plusieurs processus clés de l'entreprise. Nous avons ainsi identifié les axes stratégiques suivants :

➢ **Partage de l'information :** Nous avons permis aux collaborateurs de consulter facilement les informations sur les équipes, les départements et les managers, ainsi que sur la structure organisationnelle globale de l'entreprise.

➢ **Gestion des projets :** nous avons mis en place le Framework **Scrum**, cette approche permet une gestion itérative et incrémentielle des projets, favorisant une collaboration étroite entre les équipes et une adaptation rapide aux changements. De plus, nous avons digitalisé la planification des sprints, la création des product backlogs, ainsi que la gestion des tâches.

➢ **Gestion du patrimoine documentaire :** Nous avons développé un système de recherche rapide facilitant l'accès aux documents de l'entreprise. Cette solution centralisée facilitera la gestion, le partage et la consultation des documents, améliorant ainsi l'efficacité et la conformité.

➢ **Coordination :** Nous avons simplifié le processus de gestion des congés, des événements et d'autres activités organisationnelles. Il s'agit d'une plateforme

automatisée permettra la soumission, l'approbation et le suivi des demandes de congés, ainsi que la gestion efficace des événements internes.

➢ **Communication :** Nous avons intégré des systèmes de messagerie et de chat room pour faciliter une communication rapide et efficace entre les collaborateurs. Ces outils permettront des échanges fluides, aussi bien en temps réel qu'en mode asynchrone.

➢ **chatbot basé sur IA :** Nous avons développé un assistant virtuel sous forme de chatbot pour aider les collaborateurs à rechercher des informations sur l'organisation, à accéder aux documents, à obtenir des mises à jour sur les projets et à répondre à d'autres besoins courants de manière rapide et intuitive.

La figure (7) détaille les axes stratégiques de notre projet via le diagramme d'Ishikawa.

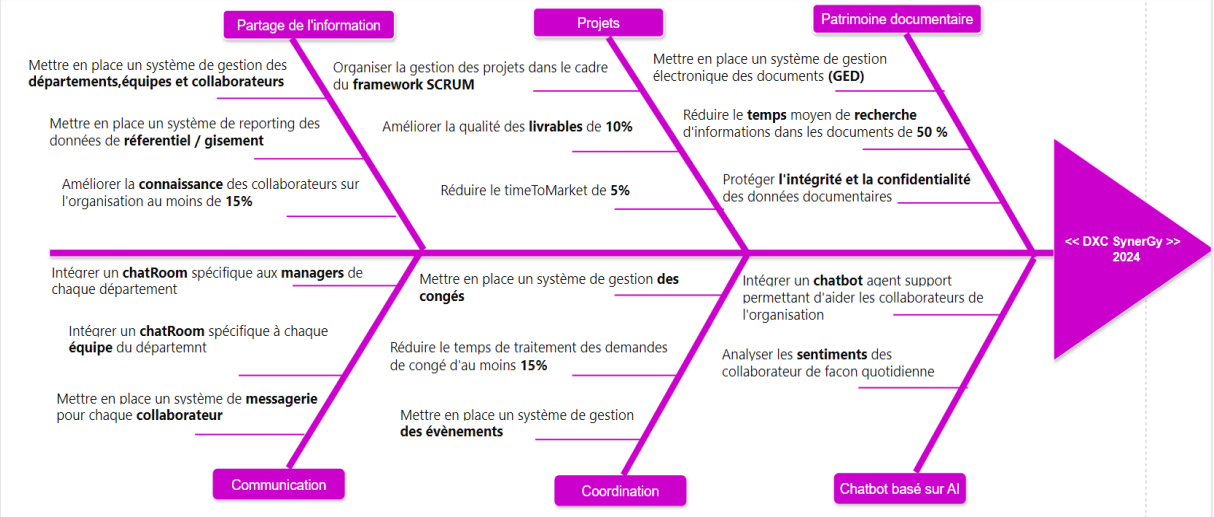


Figure 7: Diagramme d'Ishikawa

III. Méthodologie adoptée et planification du projet

Nous entamons cette phase en exposant la méthodologie adoptée durant notre stage, ainsi que la planification du projet.

III.1 Méthodologie de développement

La méthodologie de développement du projet constitue un aspect critique de sa réussite. Nous allons examiner minutieusement chaque aspect de cette méthodologie pour garantir une mise en œuvre efficace et efficiente du projet.

III.1.A Motivation du choix

Pour la conduite de notre projet, nous avons choisi d'utiliser le Framework Scrum. Ce choix s'explique par les nombreux avantages que cette méthode agile offre dans la gestion de projets complexes et dynamiques.

Scrum est reconnu pour sa flexibilité, permettant de s'adapter rapidement aux changements et aux nouvelles exigences du projet. Cette approche itérative et incrémentielle divise le projet en sprints, de courtes périodes de travail intensif, généralement de deux à quatre semaines. Chaque sprint se termine par une revue permettant d'évaluer les progrès réalisés et de s'adapter pour le sprint suivant.

La figure (8) met en évidence l'architecture du Framework Scrum. **[8]**

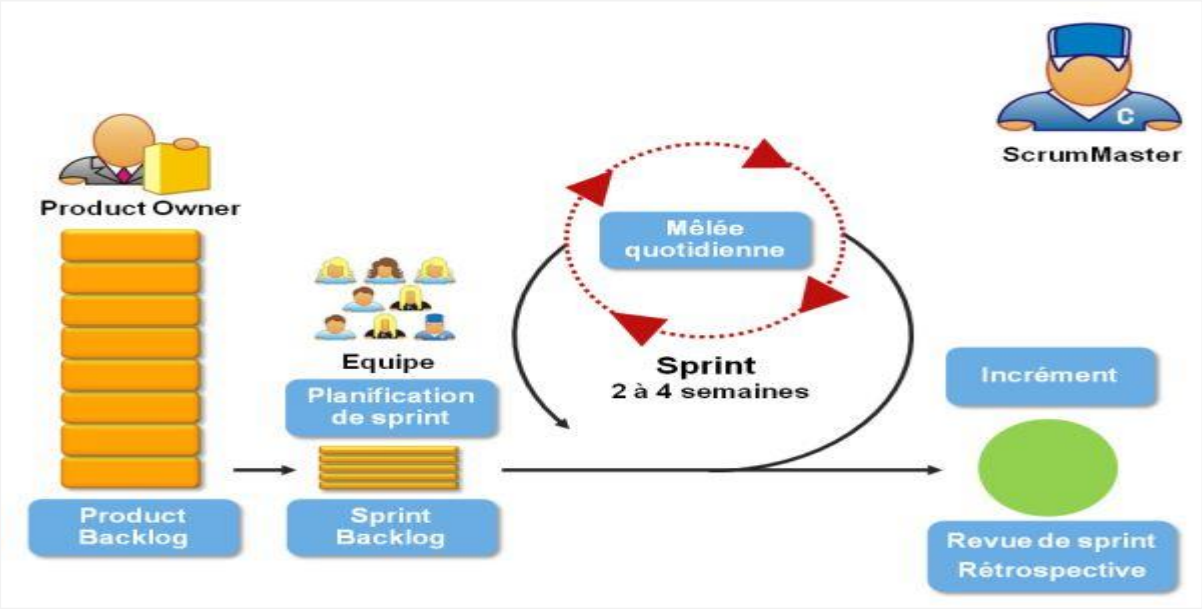


*Figure 8: Architecture du Framework Scrum **[8]***

III.1.B Répartition des rôles Scrum dans le projet

Étant donné qu'il s'agit d'un projet interne pour DXC Technology Maroc, le Product Owner est un collaborateur de l'organisation. De même pour le Scrum Master et l'équipe de développement.

Le tableau (3) illustre la répartition des rôles Scrum dans notre projet.

Rôle	*Responsable*
Product Owner	M. JILANI Omar (Manager APPS)
Scrum Master	Mme. IBRAHIMI Safae (Team Lead APPS)
Equipe de développement	M. KHOUNA Younes (Membre APPS) M. IBRAHIMI Amine (Membre APPS)

Tableau 3: Répartition des rôles Scrum dans le projet

III.2 Planification du projet

Nous débuterons cette section en présentant la planification des sprints du projet, suivie de l'estimation de sa durée totale. Enfin, nous synthétiserons ces informations à l'aide d'un diagramme de Gantt.

III.2.A Planification des sprints

Après avoir réparti les rôles clés au sein de la méthodologie Scrum, nous procéderons à la présentation des divers sprints du projet ainsi que l'ensemble des taches y sont associés.

Le tableau (4) illustre tous les sprints établis au cours de notre projet.

Sprint	*Date début*	*Date fin*	*Tâche*
Sprint 01 : Étude préalable	01/02/2024	15/02/2024	1- Faire l'étude de faisabilité. 2- Identifier les risques. 3- Estimer le cout-délai-efficacité. 4- Planifier le prochain sprint.
Sprint 02 : Analyse et spécification fonctionnelle	15/02/2024	28/02/2024	1- Exprimer les besoins fonctionnels. 2- Exprimer les besoins de la Base de données. 3- Définir des maquettes. 4- Identifier des contraintes de développement. 5- Planifier du prochain sprint.
Sprint 03 : Choix de l'architecture et conception détaillée	01/03/2024	10/03/2024	1- Valider l'architecture de Microservices pour l'application. 2-Répartir les microservices pour chaque membre de l'équipe de développement. 3- Etablir les diagrammes de cas d'utilisation, de classe, de séquence et d'activité de chaque microservice. 4-Valider des diagrammes

			5-Planifier le prochain sprint
Sprint 04 : Développement de identity-service et patrimoineDoc-service	11/03/2024	26/03/2024	1-Développer le backend du microservice identity-service et patrimoineDoc-service. 2- Faire les teste unitaire avec JUNIT & Mockito. 3-Conteneuriser des deux microservices avec Docker. 4- Inscrire les microservices dans Eureka. 5- Surveiller et Observer les deux microservices avec Zipkin. 6-Developer le frontend. 7-Valider les deux microservice. 8-Planifier du prochain sprint.
Sprint 05 : Développement du projectScrum-service,feelingBoard-service et ChatRoom	27/03/2024	15/04/2024	- Même étapes du sprint (04)
Sprint 06 : Développement du hollidays-service et privateChat-service	15/04/2024	30/04/2024	- Même étapes du sprint (04)
Sprint 07 : Développement du chatBotRAG-service et forum-service	02/05/2024	12/05/2024	- Même étapes du sprint (04)
Sprint 08 : Développement du notification-service et proxy-service	13/05/2024	29/05/2024	- Même étapes du sprint (04)

Tableau 4: Planification des sprints

III.2.B Durée totale du projet

Le cycle de développement de notre solution s'est étendu sur une période de **109 jours**, répartis en **huit (8) sprints** distincts. Chaque sprint a été initié uniquement après l'achèvement du sprint précédent, conditionné par la validation du backlog correspondant. Cela signifie que chaque phase successive du développement n'a débuté qu'après la confirmation que les objectifs et les tâches du sprint antérieur avaient été pleinement réalisés et validés.

III.2.C Outil de planification de projet : GANTT Project

Pour l'outil de planification, nous avons utilisé le logiciel GANTT Project. Ce choix a été motivé par plusieurs raisons. Tout d'abord, GANTT Project est un logiciel open-source, ce qui signifie qu'il est accessible sans frais de licence, réduisant ainsi les coûts pour l'organisation. De plus,

l'open-source offre une flexibilité et une adaptabilité accrues, permettant de personnaliser l'outil en fonction des besoins spécifiques du projet.

La figure (9) montre le diagramme de Gantt pour la planification des sprints.

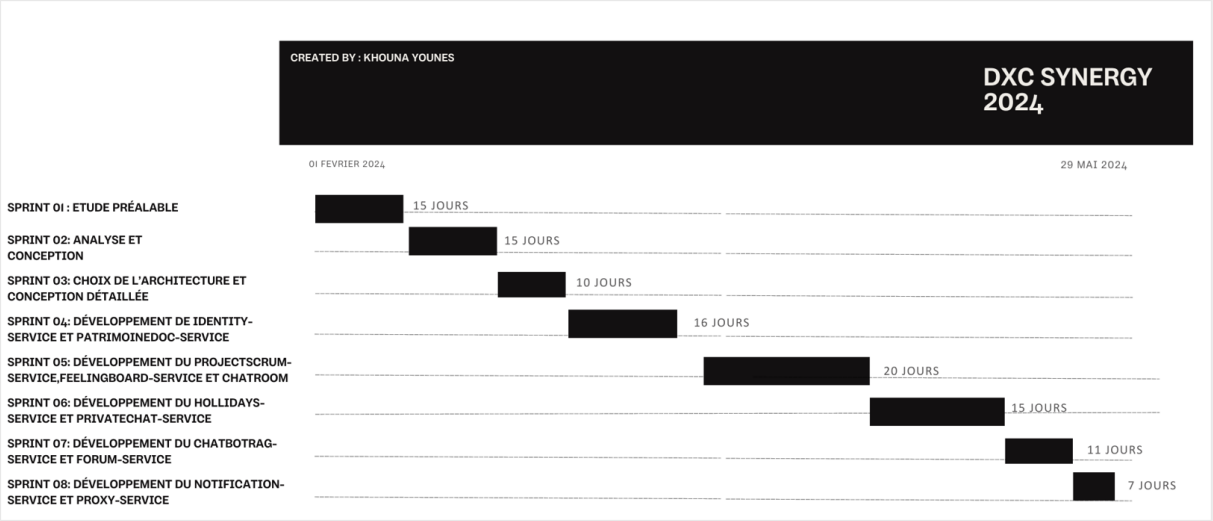


Figure 9: Diagramme de Gantt

III.3 Outil de gestion de projet : JIRA

Pour la conduite de notre projet, nous avons exploité JIRA, un outil puissant et flexible de gestion de projets. JIRA nous a permis de créer et de gérer efficacement le product-backlog, en définissant clairement les fonctionnalités, les tâches et les user stories. Ce choix a été motivé par plusieurs avantages clés que JIRA offre.

Les raisons de notre choix incluent :

➢ **Gestion centralisée :** JIRA offre une plateforme centralisée pour toutes les activités de gestion de projet, facilitant la collaboration et la communication entre les membres de l'équipe.

➢ **Suivi et Transparence :** L'outil permet un suivi précis de l'avancement des tâches et des user stories, offrant une transparence totale sur le statut du projet.

➢ **Personnalisation et Flexibilité :** JIRA est hautement personnalisable, ce qui nous permet d'adapter les workflows et les tableaux de bord selon les besoins spécifiques de notre projet.

➢ **Support et communauté :** Avec une large base d'utilisateurs et une documentation exhaustive, JIRA bénéficie d'un support solide et d'une communauté active, facilitant la résolution de problèmes et l'optimisation de son utilisation.

Les figure (10) illustre un aperçu du sprint (4) de notre projet avec l'outil JIRA.

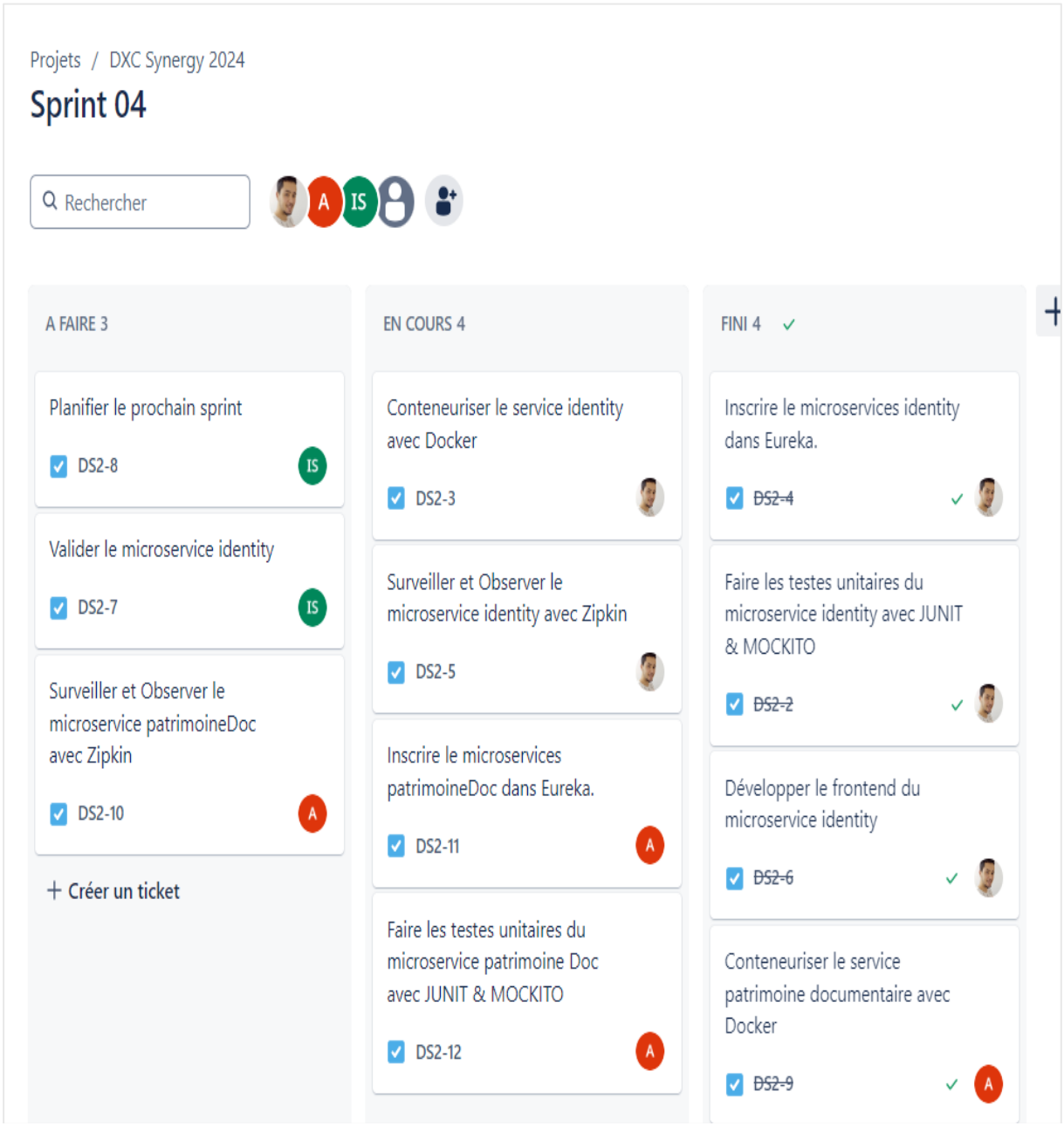


Figure 10: Aperçu du sprint (4) avec JIRA

IV. Synthèse

Dans ce premier chapitre, nous avons présenté l'organisme d'accueil, en détaillant son secteur d'activité, sa structure, ainsi que ses clients et partenaires stratégiques. Ensuite, nous avons décrit le contexte général du projet, y compris la problématique identifiée et la solution proposée. Enfin, nous avons abordé la planification du projet, en expliquant les choix méthodologiques et les outils utilisés.

La prochaine étape de notre rapport se concentre sur l'analyse et la conception. Ce chapitre crucial détaillera les exigences fonctionnelles et non fonctionnelles, les spécifications techniques et les différents modèles conceptuels développés pour le projet.

Chapitre II : Spécification fonctionnelle

Dans ce chapitre consacré à la phase de spécification fonctionnelle, nous commençons par définir le périmètre du projet. Cette étape initiale permet de d'identifier les acteurs du système. Ensuite, nous abordons les exigences fonctionnelles, qui détaillent les fonctionnalités et les comportements attendus du système. Une fois ces aspects couverts, nous passons aux exigences non fonctionnelles, qui spécifient les critères de performance, de sécurité et d'ergonomie.

Enfin, nous présentons les maquettes (ou encore la conception graphique) qui illustrent de manière visuelle et intuitive la disposition et l'interaction des éléments de l'application.

I. Périmètre du projet

Étant donné qu'il s'agit d'un projet interne pour l'organisation, l'identification des acteurs a été relativement simple et directe, facilitée par notre connaissance approfondie de la structure de l'entreprise.

Le tableau (5) met en évidence les acteurs identifiés ainsi que les permissions attribuées.

Acteur	***Permissions***
Manager	1- Gestion des départements, équipes et collaborateurs. 2- Ajouter, terminer, annuler ou mettre ne pause un projet. 3- Sélectionner ou modifier le scrum master d'un projet. 4- Traitement (approuver ou rejeter) les demandes de congé. 5- Consulter le feeling Board des collaborateurs. 6- Gestion des documents. **(*)** + Permissions attribuées au collaborateur.
Chef d'équipe (Team lead)	1 -Gestion des sprints. 2- Gestion des tâches d'un projet. **(*)** + permissions attribuées au collaborateur.
Scrum master	1- Gestion du product-backlog d'un projet. 2- gestion des fonctionnalités d'un projet. **(*)** + Permissions attribuées au collaborateur.
Collaborateur	1- Envoie ou Annulation (si la demande est non encore traitée) d'une demande de congé. 2- Envoyer un message privé ou dans le chat room de son équipe. 3- Modifier l'état de sa tâche lors d'un projet. 4- Ajouter un sentiment dans le feeling board. 5- Publier une activité dans le forum de partage. 6- Ajouter des commentaires aux activités publiées. 7- Accès au chatbot de l'application. **(*)** + Permissions attribuées au collaborateur .

Tableau 5: Acteurs du système

II. Besoins fonctionnels

Lors de la rédaction de notre Cahier des Charges Fonctionnel (CDCF) et après une étude approfondie de la faisabilité du projet, nous avons identifié les besoins fonctionnels spécifiques pour chaque module de l'application.

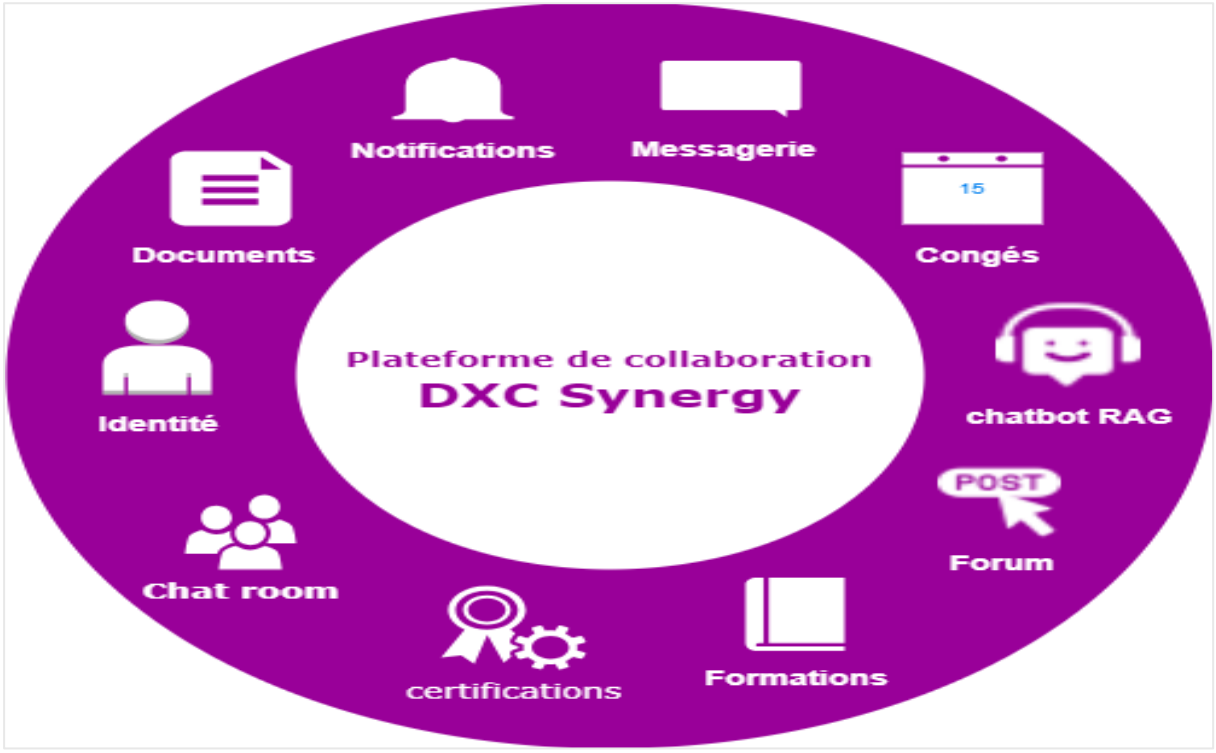


Figure 11: Modules identifiés dans notre projet

II.1 Module gestion de l'identité

Le module gestion de l'identité se concentre principalement sur l'axe stratégique cité précédemment (voir figure 7) à savoir « Partage de l'information », et doit satisfaire les exigences fonctionnelles suivantes :

➢ **Authentification :** Un collaborateur peut s'authentifier via l'adresse électronique et le mot de passe de l'organisation, ce dernier pouvant être modifié après la première connexion à l'application.

➢ **Accès à l'espace « Mon équipe » :** Un collaborateur a accès aux informations de son équipe, y compris celles de ses coéquipiers, de son chef, ainsi qu'aux statistiques de celle-ci, telles que l'état de l'équipe (en projet, disponible.), le nombre de collaborateurs, et le nombre de projets réalisés etc.

➢ **Accès à l'espace « Mon département » :** Un manager a accès aux informations de son département, y compris celles des managers, des équipes et des collaborateurs de ce dernier.

➢ **Gestion des équipes :** Après avoir consulté l'espace « Mon Département », le manager du département peut ajouter, supprimer ou modifier une équipe.

➢ **Gestion des collaborateurs** : Apres avoir consulter une équipe de son département, le manager peut ajouter un nouveau collaborateur, supprimer ou encore le modifier.

II.2 Module gestion du patrimoine documentaire

Ce module se concentre principalement sur l'axe stratégique « Patrimoine documentaire », tout en abordant également l'axe « Partage de l'information ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Accès à l'espace « Documents » :** Un collaborateur peut consulter les documents de référentiel de l'organisation ainsi il peut les télécharger.

➢ **Gestion des documents :** Un manager a la possibilité de téléverser de nouveaux documents tout en précisant les niveaux d'accès, notamment les documents publics, privés pour un département, privés pour une équipe, ou encore privés pour un collaborateur spécifique.

II.3 Module gestion des projets

Ce module ou encore le microservice se concentre principalement sur l'axe stratégique « Projets », tout en abordant également l'axe « Coordination ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Accès à l'espace « Projets » :** Un collaborateur a la possibilité de consulter l'état d'avancement du projet en cours affecté à son équipe. Il peut ainsi accéder aux informations telles que le Scrum master du projet et les équipes affectées à celui-ci.

➢ **Affectation des projets :** Un manager peut ajouter un nouveau projet et préciser le Scrum master qui va s'occuper de la conduite de ce dernier.

➢ **Ajout du product-backlog :** Après avoir été assigné à un projet, le Scrum Master sera chargé d'ajouter toutes les fonctionnalités requises dans le Product Backlog. De plus, il aura la possibilité de sélectionner les équipes qui travailleront sur ce projet.

➢ **Planification des sprints :** Après avoir assigné son équipe à un projet, le chef de celle-ci (Team Lead) peut planifier un sprint en spécifiant la date de début, la date de fin et une description détaillée.

➢ **Affectation des tâches :** Au début d'un sprint, le chef d'équipe ajoutera toutes les tâches à réaliser, tout en les attribuant aux membres de son équipe. Il précisera la priorité, la complexité ainsi que les dates de début et de fin de chaque tâche.

➢ **Modification de l'état d'une tâche :** Lorsqu'un collaborateur termine une tâche, il a la possibilité de la marquer comme terminée. Cette action peut également être effectuée par son chef d'équipe.

➢ **Consultation de la Burndown chart :** le Scrum master et les chef d'équipes peuvent voir la Burndown-chart(proportion tâches restante réelles / tâches restantes estimées) du sprint en cours.

II.4 Module traitement des demandes de congé

Ce module se concentre principalement sur l'axe stratégique « Coordination ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Accès à l'espace « Congés » :** Les collaborateurs ont la possibilité d'envoyer une demande de congé ou de suivre l'état d'avancement de celle-ci s'il est déjà envoyé. De plus, ils ont la faculté de l'annuler si elle n'a pas encore été traitée. Ils peuvent également consulter quelques statistiques relatives aux congés de leur équipe ainsi que l'historique de leurs propres demandes.

➢ **Déclarer un remplaçant (backup) :** Un collaborateur doit obligatoirement mentionner son remplaçant pour sa demande de congé En l'absence de cette information, la demande ne pourra être transmise.

➢ **Traitement des demandes :** Un manager peut approuver ou rejeter une demande de congé d'un collaborateur.

II.5 Module feeling-board

Ce module se concentre principalement sur l'axe stratégique « Communication ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Publication d'un sentiment :** Les collaborateurs ont la possibilité de partager leur sentiment du jour en choisissant parmi trois options : Satisfait, Neutre, Insatisfait. Ils peuvent également laisser un feedback pour justifier leur choix, cette étape étant facultative.

➢ **Consultation du feeling-board :** Le chef d'équipe a accès au tableau des sentiments (feeling-board) des membres de son équipe. De même, le manager peut le consulter pour l'ensemble des collaborateurs de son département.

II.6 Module forum de partage

Ce module se concentre principalement sur l'axe stratégique « Communication ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Publication d'une activité :** Un collaborateur a la possibilité de partager une activité, un événement marquant ou une formation obtenue sur le forum de partage.

➢ **Suppression d'une activité :** Un collaborateur peut supprimer sa publication.

➢ **Modification d'une activité :** un collaborateur peut à tout instant modifier la description ou même l'image de son activité publiée.

➢ **Ajout d'un commentaire :** Un collaborateur peut commenter une activité publiée par son collègue.

II.7 Module chat room

Ce module se concentre principalement sur l'axe stratégique « Communication ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Envoie d'un message :** Chaque collaborateur est automatiquement intégré au chat room de son équipe, lui permettant ainsi de communiquer avec ses coéquipiers par le biais de messages textuels simples ou en partageant des photos.

➢ **Réaction à un message :** À tout moment, un collaborateur peut réagir à un message envoyé par un membre de son équipe.

II.8 Module messagerie

Ce module se concentre principalement sur l'axe stratégique « Communication ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Envoie d'un message privé :** Un collaborateur peut envoyer des messages privés à d'autres utilisateurs de l'application par le biais de messages textuels simples ou en partageant des photos.

➢ **Réaction à un message :** À tout moment, un collaborateur peut réagir à un message reçu.

II.9 Module chatbot RAG

Ce module se concentre principalement sur l'axe stratégique « chatbot basé sur IA ». Il est chargé de répondre aux exigences fonctionnelles suivantes :

➢ **Accès au chatbot :** Notre chatbot est conçu pour répondre aux questions posées par les collaborateurs, à condition qu'elles restent dans le cadre du contexte fourni au bot. Si une question est posée en dehors de ce contexte, le chatbot ne pourra pas y répondre.

➢ **Mise à jour :** Un manager a la possibilité de mettre à jour le contexte du chatbot en téléchargeant manuellement des documents au format PDF, JPG, DOCX. Cette approche est réalisée selon le modèle RAG (Retrieval Augmented Generation).

II.10 Module notification

Le module de notification se concentre principalement sur l'axe stratégique de la communication. Il est chargé de notifier les collaborateurs de tous les événements qui les concernent, tels que le traitement des demandes de congé, l'attribution de nouveaux projets, de nouvelles tâches ou de nouveaux documents.

III. Besoins non fonctionnels

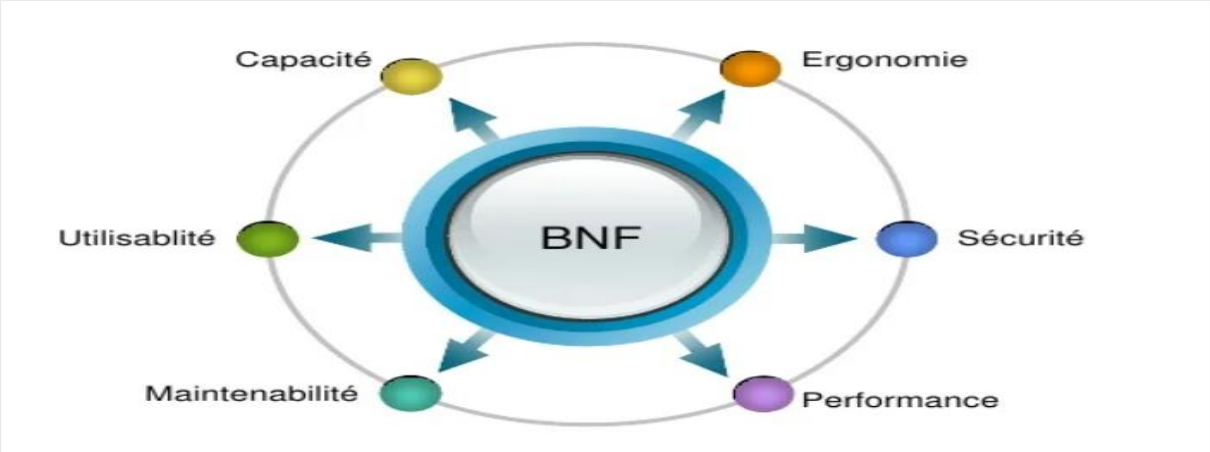


Figure 12: Besoins non fonctionnels ***[9]***

Les besoins non fonctionnels d'un site web concernent des aspects qui ne sont pas directement liés aux fonctionnalités spécifiques, mais qui sont essentiels pour garantir des performances, une sécurité, une convivialité et une qualité globale.

III.1 Point de vue utilisateur

➢ **Performance :** Notre application a été conçu pour offrir une expérience utilisateur fluide et sans interruption. Grâce à des techniques de mise en cache et de minification des ressources, les pages se chargent en moins de trois secondes, garantissant ainsi une interaction rapide et efficace. Les actions de l'utilisateur, telles que les clics et les défilements, reçoivent une réponse quasi instantanée, avec un **temps de réaction** de moins de **100 millisecondes,** grâce à l'optimisation JavaScript.

➢ **Sécurité :** Nous avons stocké les données sensibles de manière sécurisée grâce à des techniques de cryptage avancées, assurant leur confidentialité et leur intégrité. En particulier, les mots de passe des utilisateurs sont cryptés avec **Bcrypt**, une méthode éprouvée et sécurisée pour protéger les informations sensibles. Pour prévenir les attaques de scripts intersites (XSS), nous appliquons des mesures de sécurité strictes, incluant la validation des entrées et l'encodage des sorties. En outre, nous avons mis en place des mécanismes d'authentification robustes, y compris l'authentification à jeton avec **JWT** (Json Web Token).

➢ **Intégrité :** Notre application se distingue par son engagement à garantir l'intégrité des données, offrant une protection robuste contre les intrusions et les accès non autorisés. Chaque **module** de l'application dispose de **sa propre base de données**, renforçant ainsi la sécurité et la gestion des données. Nous utilisons des techniques de cryptage avancées pour protéger les données sensibles, tant en transit qu'au repos, assurant ainsi leur confidentialité et leur intégrité.

➢ **Ergonomie :** Nous avons conçu design minimaliste qui est à la fois esthétiquement plaisant et fonctionnel, facilitant la navigation et l'interaction. Les menus et les options sont organisés de manière logique, permettant aux utilisateurs de **trouver rapidement** ce qu'ils cherchent sans être submergés par des éléments superflus. Nous avons également intégré des fonctionnalités de personnalisation, permettant aux utilisateurs d'adapter l'interface **à leurs préférences**.

III.2 Point de vue développeur

➢ **Évolutivité :** Nous avons opté pour une **architecture microservices**. Cela implique que chaque composant de notre application est conçu de manière indépendante, offrant ainsi une flexibilité considérable lors de l'introduction de nouvelles fonctionnalités.

➢ **Résilience :** Nous avons mis en œuvre des pratiques et des mécanismes qui favorisent la résilience de notre système. Tout d'abord, nous avons mis en place une **architecture distribuée** qui répartit la charge de travail sur plusieurs serveurs, ce qui réduit les risques de points de défaillance uniques.

➢ **Testabilité :** Nous avons accordé une attention particulière à la testabilité de notre application, en mettant en place un ensemble complet de pratiques et de techniques de test. Cela comprend notamment la mise en œuvre **de tests unitaires** pour chaque composant logiciel, permettant de vérifier le bon fonctionnement des unités de code individuelles. De plus, nous avons réalisé des **tests d'intégration** pour évaluer la cohérence et l'interaction entre les différents modules de l'application.

➢ **Maintenabilité :** La maintenabilité de notre application est assurée par notre choix d'architecture basée sur les microservices. Cette approche modulaire divise notre système en plusieurs composants indépendants, chacun se concentrant sur une fonctionnalité spécifique de l'application.

III.3 Point de vue communication

➢ **Interopérabilité :** nous avons choisi d'adopter une architecture basée sur la **communication asynchrone** et événementielle. Cette approche, connue sous le nom d'architecture orientée événements, permet une interaction flexible entre les différents composants de notre système. Plutôt que de dépendre d'une communication synchrone où les composants attendent activement une réponse immédiate, notre architecture permet à chaque composant d'émettre des événements lorsqu'une action significative se produit. Ces événements sont ensuite traités de manière asynchrone par d'autres composants qui y sont abonnés, ce qui permet une exécution efficace des tâches sans bloquer les ressources.

➢ **Portabilité :** Pour garantir une portabilité maximale, nous avons intégré la **conteneurisation avec Docker** dans notre architecture. En encapsulant chaque composant de notre application et ses dépendances dans des conteneurs Docker, nous assurons que notre application peut être déployée de manière cohérente et fiable sur n'importe quel environnement, qu'il s'agisse d'un ordinateur portable local, d'un serveur en nuage ou d'un cluster Kubernetes.

➢ **Réutilisabilité :** Notre application est conçue pour favoriser la réutilisabilité du code et des fonctionnalités. Nous avons adopté une approche modulaire dans la conception de notre système, en identifiant et en isolant **les composants réutilisables** qui peuvent être utilisés dans différents contextes. Ces composants sont conçus de manière à être génériques et indépendants du contexte, ce qui les rend facilement réutilisables dans d'autres parties de l'application ou dans d'autres projets.

➢ **Documentation :** Nous avons documenté chaque microservice de manière exhaustive, garantissant ainsi une compréhension claire et approfondie de l'ensemble du système. Pour chaque microservice, la documentation comprend une description détaillée de son rôle et de ses responsabilités.

IV. Maquettes

Dans cette section, nous allons présenter des maquettes conçues pour quelques modules de notre projet. Élaborées via le logiciel **Balsamiq**, ces maquettes sont validées par le client, représenté par le Product Owner, afin de garantir qu'elles répondent aux attentes et besoins définis.

IV.1 Maquette de la page d'authentification

Notre maquette d'authentification, au design minimaliste, comporte deux champs essentiels à savoir le champ « username », qui correspond à l'adresse Outlook de l'utilisateur, et le champ « password », fourni par l'organisation. Un bouton "Se connecter" permet de valider les informations saisies et d'accéder au système. Cette interface est conçue pour être simple et intuitive, assurant une expérience utilisateur fluide et sécurisée.

La figure (13) illustre la maquette d'authentification.

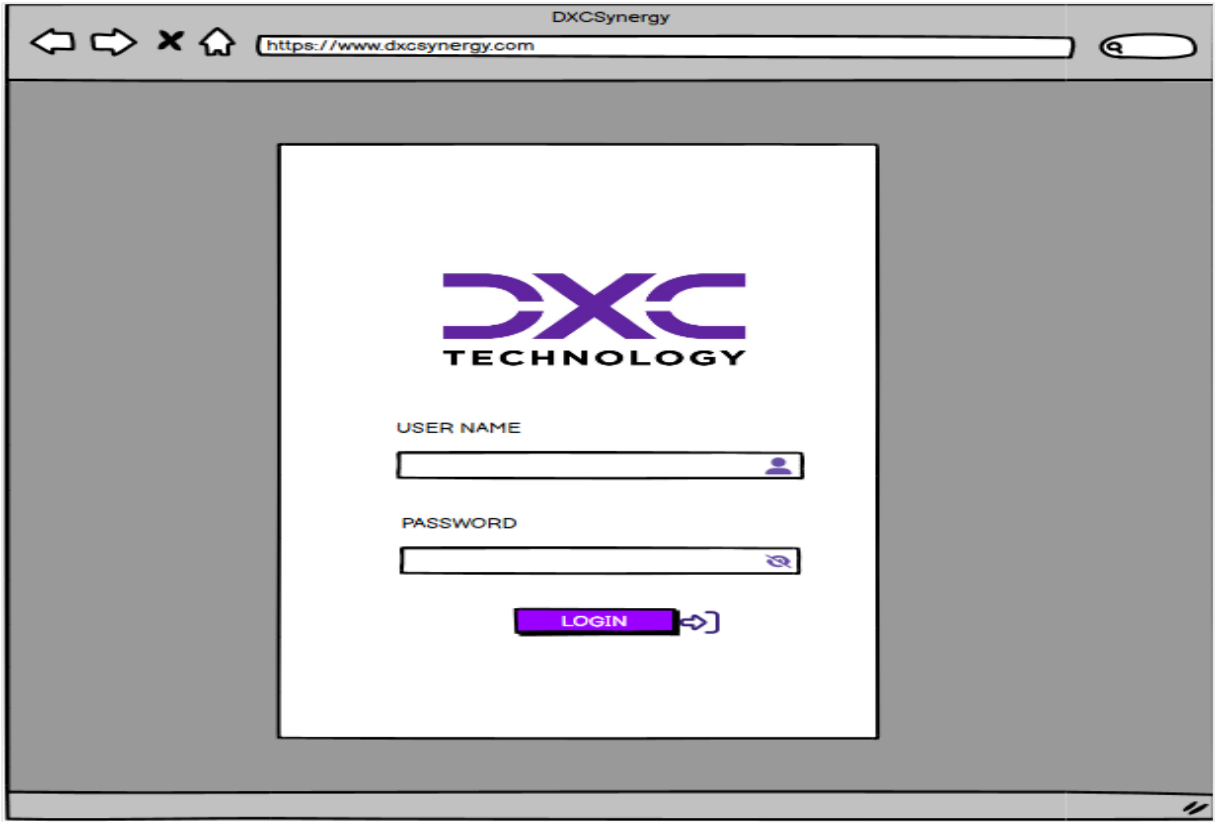


Figure 13: Maquette de la page d'authentification

IV.2 Maquette de la page d'ajout d'un nouveau collaborateur

Notre maquette de la page d'ajout d'un nouveau collaborateur, au design minimaliste, comporte les principaux champs suivants à savoir « Nom », « Prénom », « Email », « Téléphone » et « Équipe ». Un bouton « Ajouter » permet de valider les informations saisies et d'enregistrer le nouveau collaborateur dans le système.

La figure (14) montre la maquette d'ajout d'un nouveau collaborateur.

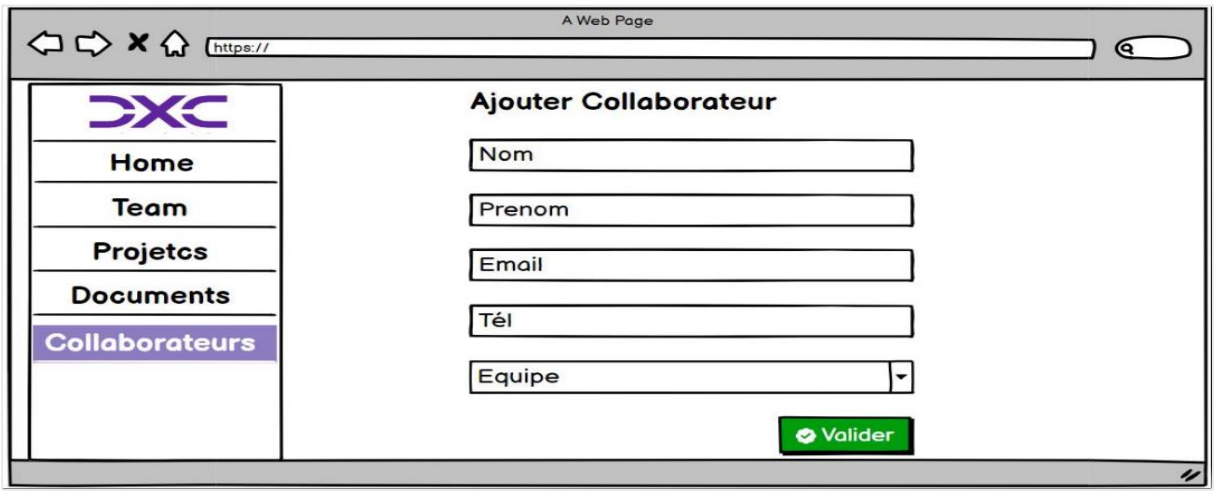


Figure 14: Maquette de la page d'ajout d'un collaborateur

IV.3 Maquette de la page de gestion des documents

Notre maquette de gestion des documents, au design minimaliste, permet d'effectuer les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer). Les champs essentiels incluent « Titre », « Thème » et « Description ». De plus, nous avons intégré des paginations pour améliorer l'expérience utilisateur (UX), facilitant ainsi la navigation et la gestion efficace des documents.

La figure (15) montre la maquette de gestion des documents.

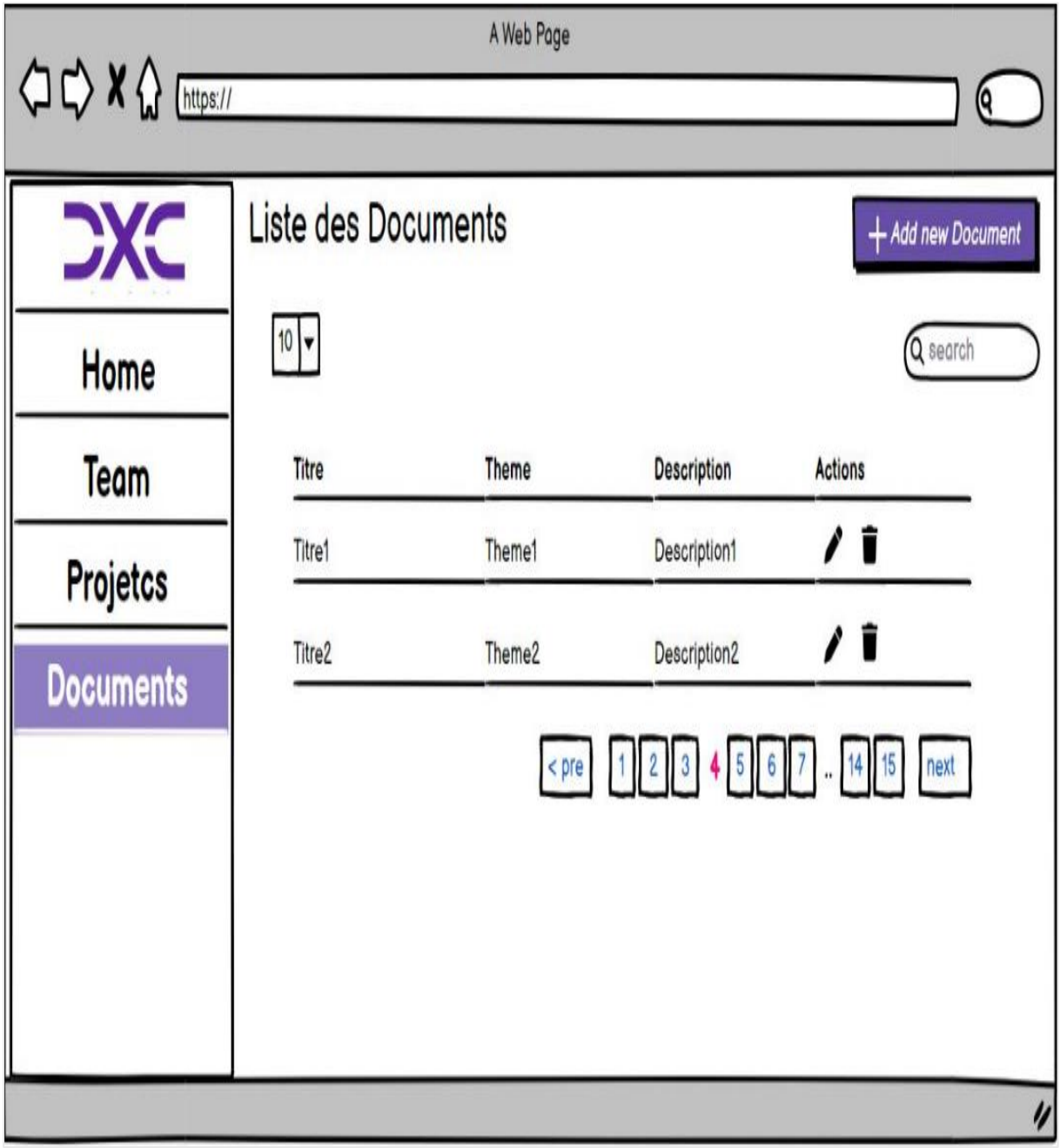


Figure 15: Maquette de la page de gestion des documents

IV.4 Maquette de la page de progression des tâches (vue chef d'équipe)

Notre maquette de progression des tâches, vue chef d'équipe, présente un design minimaliste et intuitif. Elle affiche l'état des tâches de chaque membre de l'équipe, avec trois états distincts : « À faire », « En cours » et « Terminée ». Cette interface permet au chef d'équipe de suivre facilement l'avancement des tâches et de gérer efficacement les projets en cours.

La figure (16) montre la maquette de la progression des tâches (vue chef d'équipe).

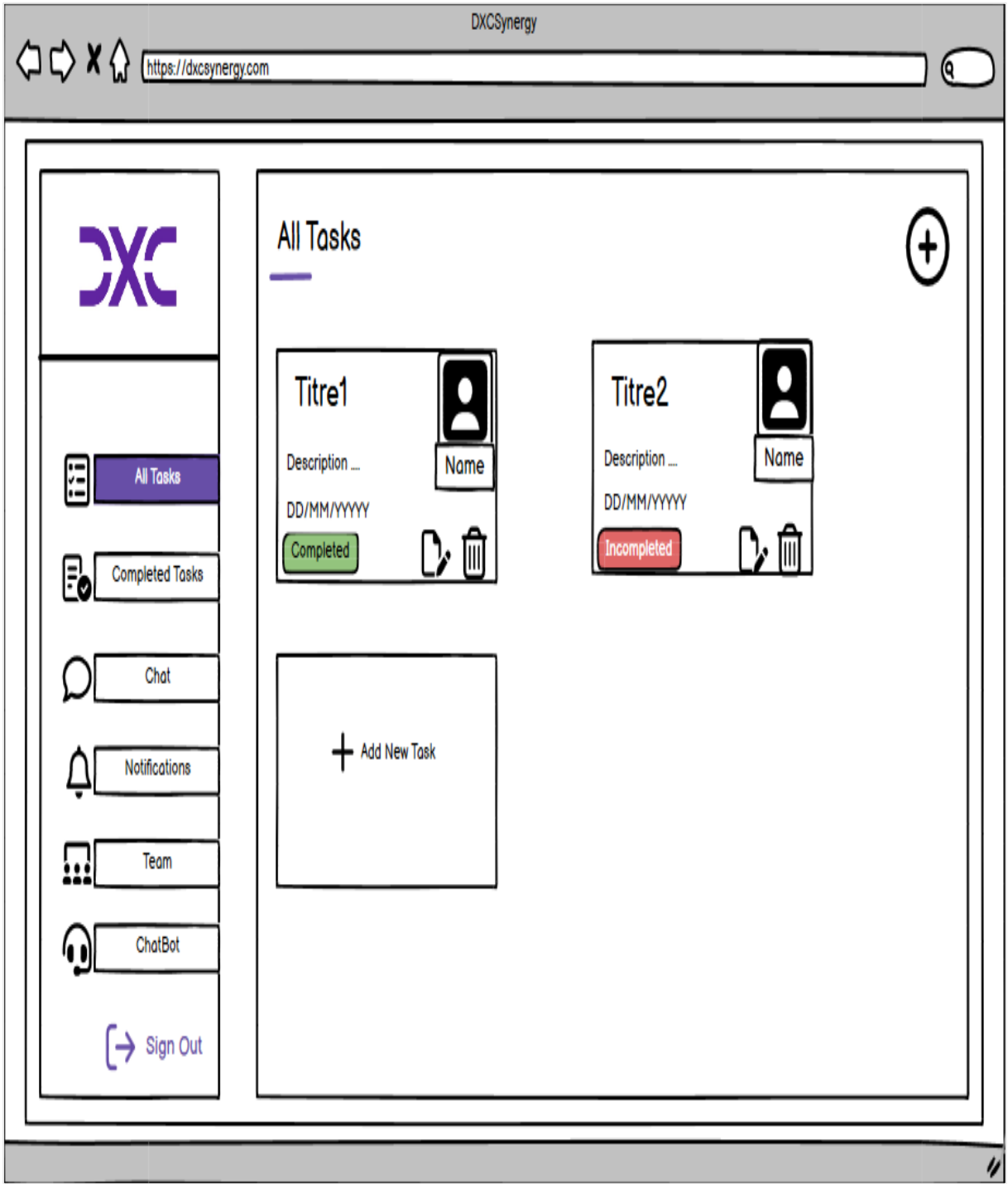


Figure 16: Maquette de la page de progression des tâches vue chef d'équipe

IV.5 Maquette de la page de progression des tâches (vue collaborateur)

Notre maquette de progression des tâches, vue chef d'équipe, présente un design minimaliste et intuitif. Elle affiche l'état des tâches de chaque membre de l'équipe, avec trois états distincts : « À faire », « En cours » et « Terminée ». Cette interface permet au chef d'équipe de suivre facilement l'avancement des tâches et de gérer efficacement les projets en cours.

La figure (17) montre la maquette de la progression des tâches (vue collaborateur).

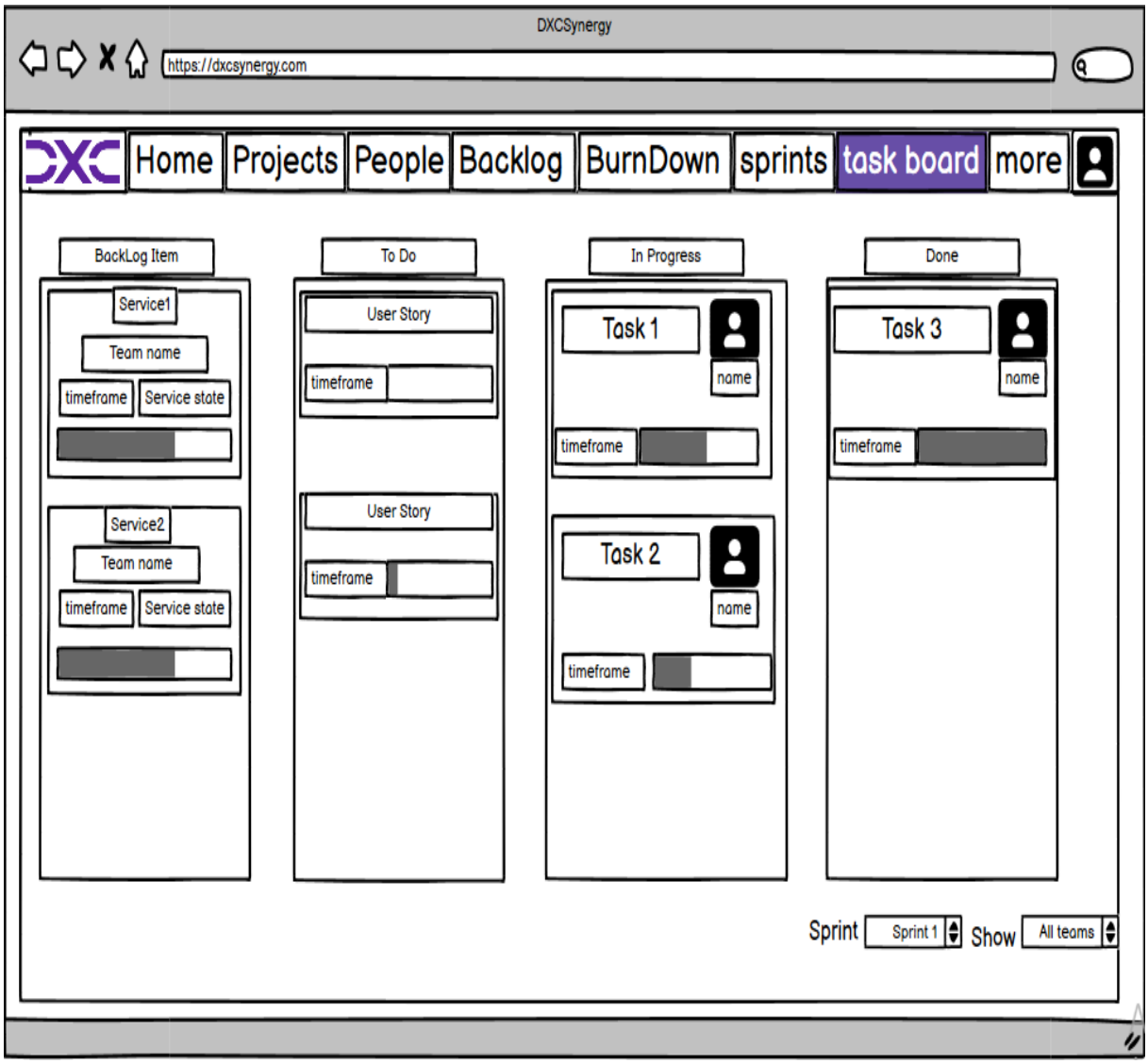


Figure 17: Maquette de la page de progression des tâches vue collaborateur

V. Synthèse

Dans cette section sur la spécification fonctionnelle, nous avons suivi une méthodologie structurée. Nous avons débuté par définir le périmètre du projet, établissant ainsi ses limites et ses acteurs. Ensuite, nous avons abordé les exigences fonctionnelles, détaillant les différentes fonctionnalités et comportements attendus du système. Nous avons ensuite examiné les exigences non fonctionnelles, couvrant les aspects de performance, de sécurité et d'ergonomie. Enfin, nous avons présenté les maquettes, offrant une représentation visuelle et intuitive de l'interface utilisateur.

La prochaine phase sera consacrée à la conception du système tout en élaborant les différents diagrammes UML.

Chapitre III : Conception (vue fonctionnelle)

Après avoir réalisé la spécification fonctionnelle des différents modules de notre application, dans cette phase, nous entamons la phase de conception de notre système et plus précisément la vue fonctionnelle du système à travers un diagramme de cas d'utilisation pour chaque module de notre système.

I. Motivation du choix du langage de conception

La décision d'utiliser **UML** pour la conception de notre projet repose sur sa capacité à fournir plusieurs perspectives essentielles. UML nous offre une variété de vues, notamment la vue fonctionnelle, structurelle et dynamique (comportementale).

Ces différentes vues nous permettent d'analyser et de modéliser le système sous différents angles, ce qui favorise une compréhension approfondie de ses aspects fonctionnels, structurels et comportementaux. Ainsi, il nous donne une vue transversale sur le système, nous permettant de comprendre les interactions entre ses différents composants et de garantir une conception cohérente et complète.

En utilisant UML, nous pouvons capturer les besoins des utilisateurs, définir l'architecture du système et spécifier son comportement de manière claire et précise.

La figure (18) montre les différentes vues du langage UML ainsi que les diagrammes correspondants**. [11]**

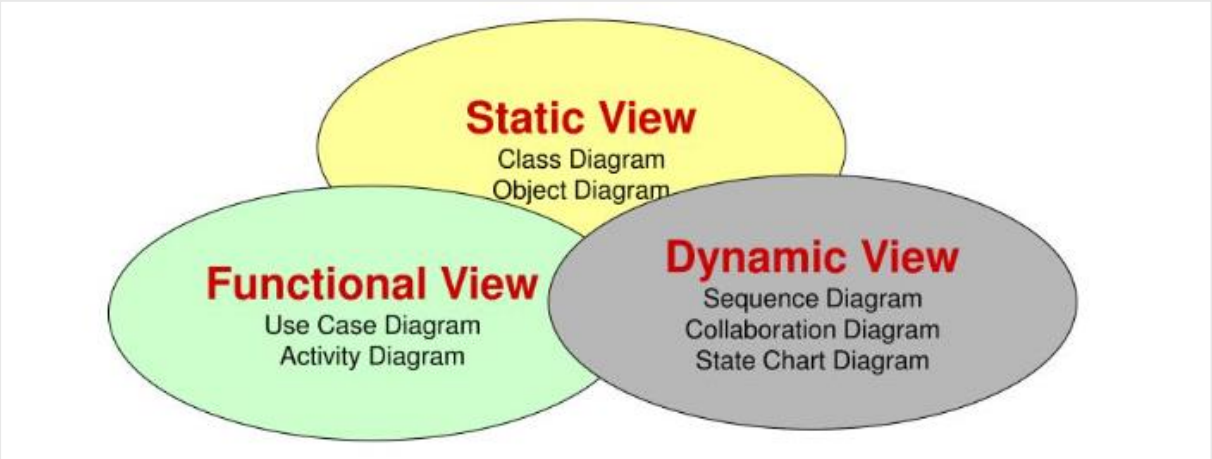


*Figure 18: Différentes Vues UML **[12]***

II. Diagramme de cas d'utilisation

Compte tenu de la modularité de notre architecture, nous avons élaboré un diagramme de cas d'utilisation pour chaque module (microservice) correspondant.

Pour réduire la complexité du diagramme de cas d'utilisation et mettre en avant les principaux pôles de notre projet, nous avons effectué un **découpage fonctionnel.** Cela signifie que nous avons créé des paquetages (package) supplémentaires pour chaque cas d'utilisation nécessitant cette approche, ce qui favorise ainsi la communication avec les intervenants

II.1 Module Gestion de l'identité.

Pour le microservice <<Identity-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (6).

Cas d'utilisation	*Acteur*	*Description*	*Lien avec autre cas d'utilisation*
S'authentifier	Tout utilisateur de l'application.	Tout utilisateur de l'application doit s'authentifier. Une fois cette étape accomplie, un jeton (token) lui est attribué, définissant son rôle et ses permissions.	N/A
Consulter information département	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur peut consulter les informations de son département, y compris l'état, les statistiques, et l'ancienneté des membres etc.	Inclut <<S'authentifier>>
Consulter informations équipe	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur peut accéder à l'espace <<Mon équipe>> ainsi il peut consulter les informations de son équipe, telles que l'état, les statistiques, et l'ancienneté etc.	Inclut <<S'authentifier>>
Consulter informations personnelles	Tout utilisateur de l'application.	Une fois accédé à l'espace <<Mon équipe>>, un collaborateur peut voir les informations essentielles sur son département.	Inclut <<S'authentifier>>
Mettre à jour informations	Tout utilisateur de l'application.	Après avoir consulté son profil, un collaborateur peut changer ses informations telle que sa situation familiale, son lien de LinkedIn etc.	Inclut <<S'authentifier>>
Gérer équipes	Manager.	Une fois authentifié, un manager peut accéder à l'espace <<Mon département>>, il aura la possibilité d'ajouter, supprimer ou encore modifier une équipe.	Inclut <<S'authentifier>>
Gérer collaborateurs	Manager.	Une fois accédé à l'espace <<Mon département>>, un	Inclut <<S'authentifier>>

		manager peut choisir une telle équipe et choisir un membre de cette dernière et le modifier ou supprimer, ou même ajouter un nouveau membre à cette équipe.	

Tableau 6: Description des cas d'utilisation de <<Identity-service>>

La figure (19) illustre le diagramme de cas d'utilisation de ce module.

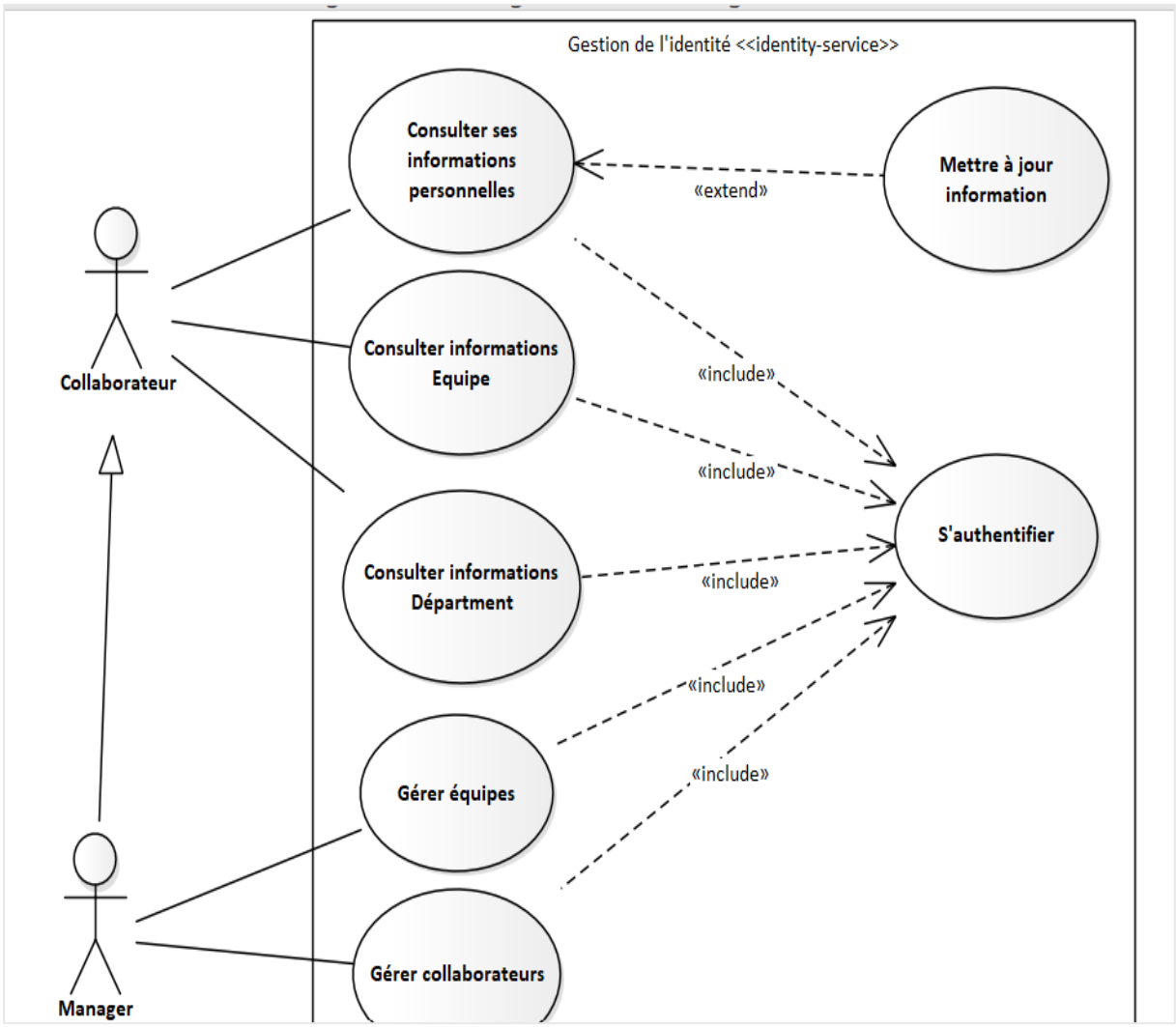


Figure 19: Diagramme de cas d'utilisation <<identity-service>>

Pour les cas d'utilisation <<Gérer collaborateurs>> et <<Gérer équipes>> sont détaillés par un package correspondant comme indiqué respectivement dans les figures (20) et (21).

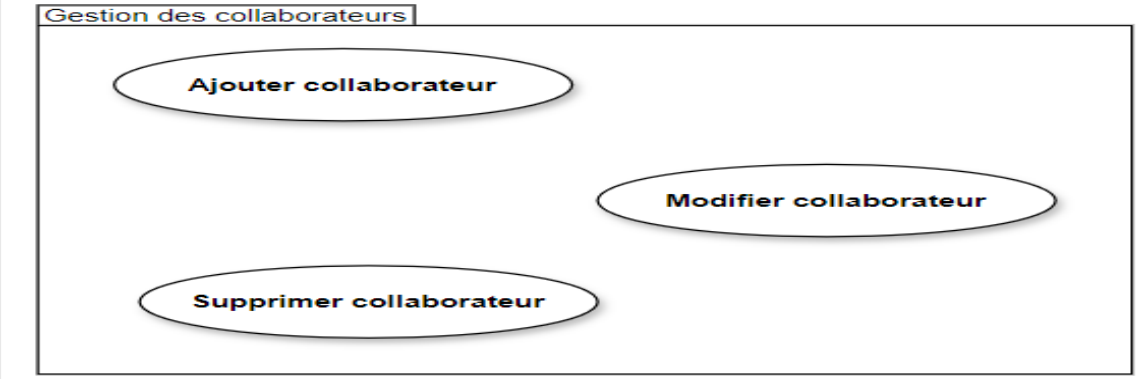


Figure 20: Paquetage cas d'utilisation <<Gérer collaborateurs>>

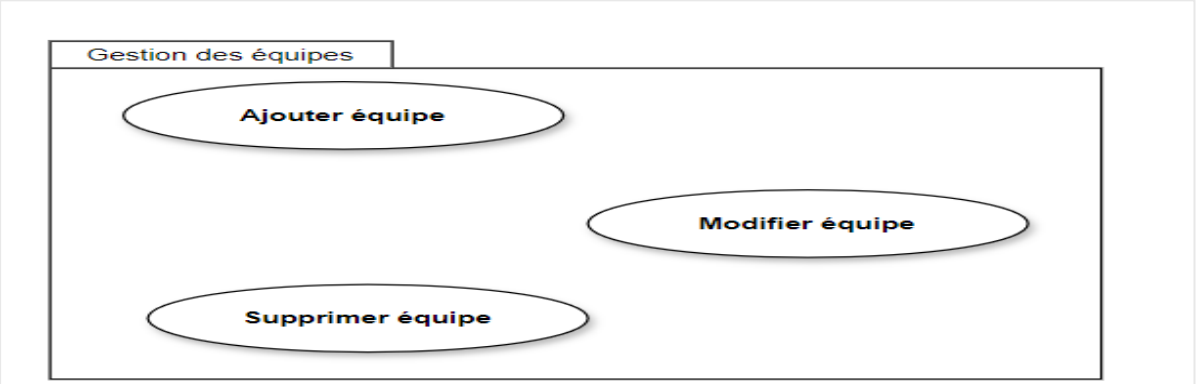


Figure 21: Paquetage cas d'utilisation <<Gérer équipes>>

II.2 Module Gestion du patrimoine documentaire

Pour le microservice <<patrimoineDoc-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (7).

Cas d'utilisation	***Acteur***	***Description***	***Lien avec autre cas d'utilisation***
Consulter document	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur peut accéder à l'espaces <<Documents>> et consulter tous les documents autorisés.	Inclut <<S'authentifier>>
Télécharger document	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur peut accéder à l'espace <<Documents>> et télécharger tous les documents autorisés.	Inclut <<S'authentifier>>
Ajouter commentaire au document	Tout utilisateur de l'application	Après avoir consulté un document, un collaborateur	Inclut <<S'authentifier>>

		peut ajouter un commentaire à ce dernier.	
Gérer documents	Manager.	Après avoir accédé à l'espaces<<Documents>> , un manager peut ajouter run nouveau document, modifier ou supprimer un document existant.	Inclut <<S'authentifier>>

Tableau 7: Description des cas d'utilisation de <<patrimoineDoc-service>>

La figure (22) illustre le diagramme de cas d'utilisation de ce module.

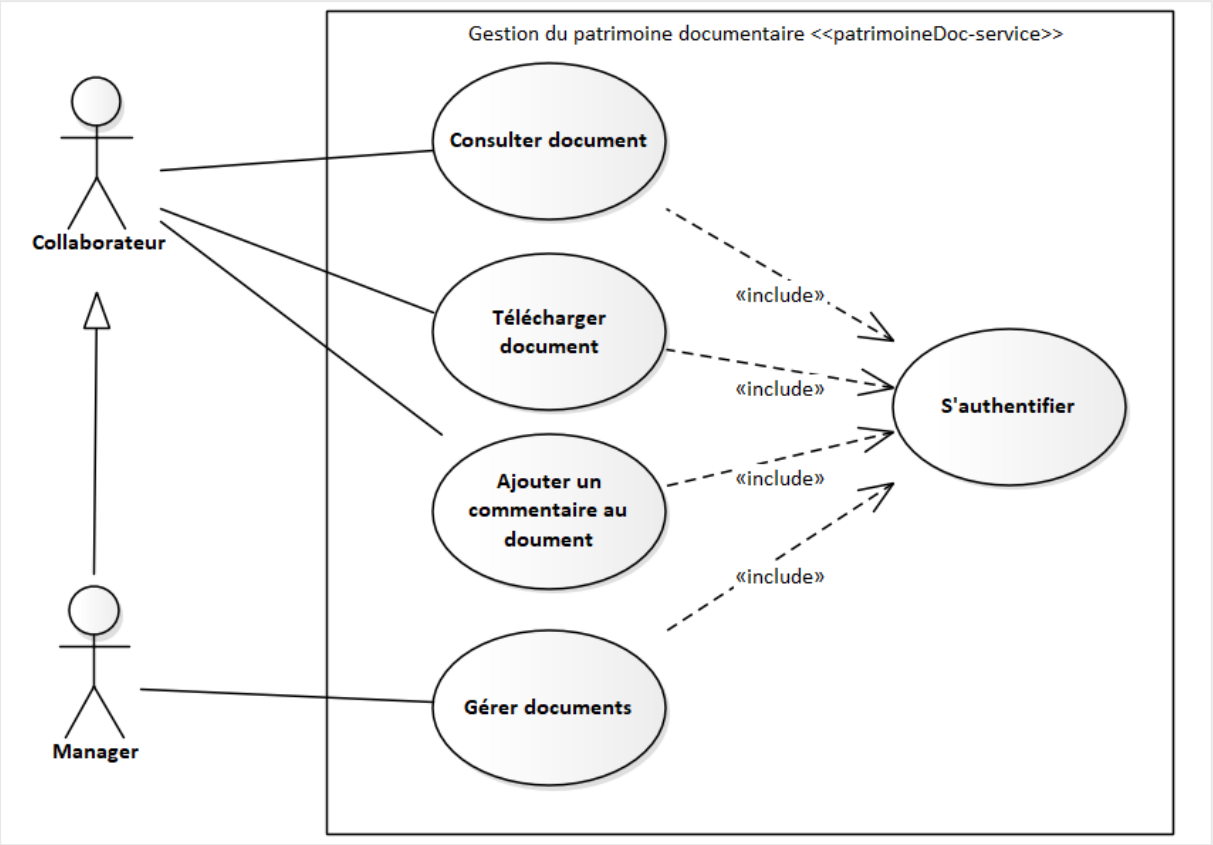


Figure 22: Diagramme de cas d'utilisation <<patrimoineDoc-service>>

Pour le cas d'utilisation <<Gérer documents>> est détaillé via un paquetage comme indiqué dans la figure (23).

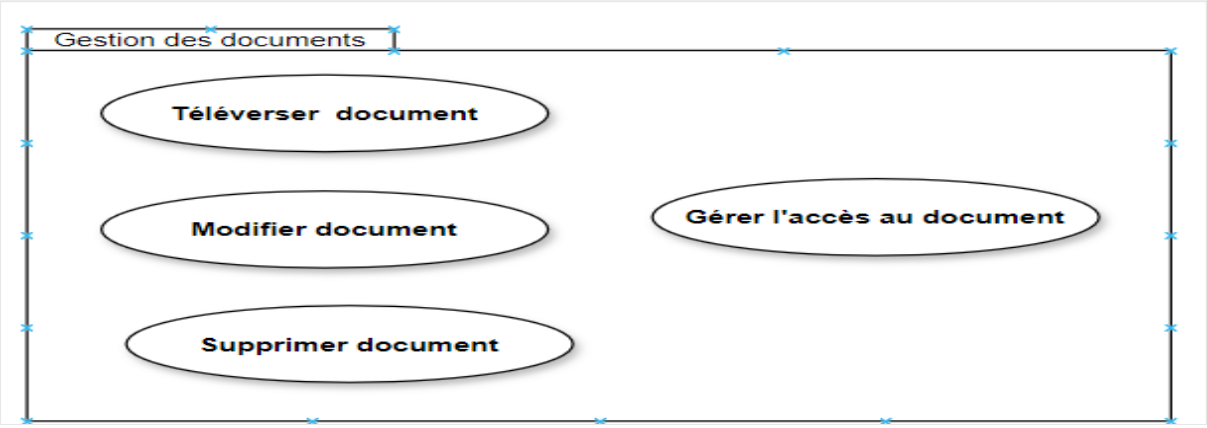


Figure 23: Paquetage cas d'utilisation <<Gérer documents>>

II.3 Module Gestion des projets

Pour le microservice <<projectScrum-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (8).

Cas d'utilisation	***Acteur***	***Description***	***Lien avec autre Cas d'utilisation***
Ajouter projet	Manager	Une fois authentifié, un manager peut accéder à l'espace <<Projets>> et ajouter un nouveau projet en spécifiant le titre, la date début, date fin et une description de ce dernier.	Inclut <<S'authentifier>>
Terminer projet	Manager	Une fois authentifié, un manager peut marquer un projet comme fini s'il voit que les fonctionnalités implémentées répondent au cahier des charges.	Inclut <<S'authentifier>>
Gérer product-backlog	Scrum master	Une fois authentifié, le scrum master du projet concerné peut accéder à l'espace <<Projets>> et gérer le product-backlog en ajoutant les fonctionnalités requis.	Inclut <<S'authentifier>>
Sélectionner équipes	Scrum master	Après avoir créé le product-backlog, le scrum master doit sélectionner les équipes concernées pour le projet en question.	Inclut <<S'authentifier>>
Gérer tâches	Chef d'équipe	Une fois authentifié, un chef d'équipe peut accéder à	Inclut <<S'authentifier>>

		l'espace <<Projets>> et gérer les tâches de son équipe.	
Consulter burndown chart	Chef d'équipe, Scrum master	Pour le chef d'équipe, il peut consulter le burndown chart des **tâches** accomplis lors d'un sprint. Pour le scrum master, il peut consulter le burndown chart des **fonctionnalités implémentés** lors d'un sprint.	Inclut <<S'authentifier>>
Planifier sprints	Chef d'équipe	Une fois accédé à l'espace <<Projets>>, un chef d'équipe peut planifier un sprint pour son équipe tout en spécifiant la date début et fin.	Inclut <<S'authentifier>>
Terminer tâche	Collaborateur et chef d'équipe.	Une fois authentifié et après avoir compléter sa tâche, un collaborateur peut marquer sa tache come terminé dans l'application.	Inclut <<S'authentifier>>

Tableau 8: Description des cas d'utilisation de <<projectScrum-service>>

La figure (24) illustre le diagramme de cas d'utilisation de ce module.

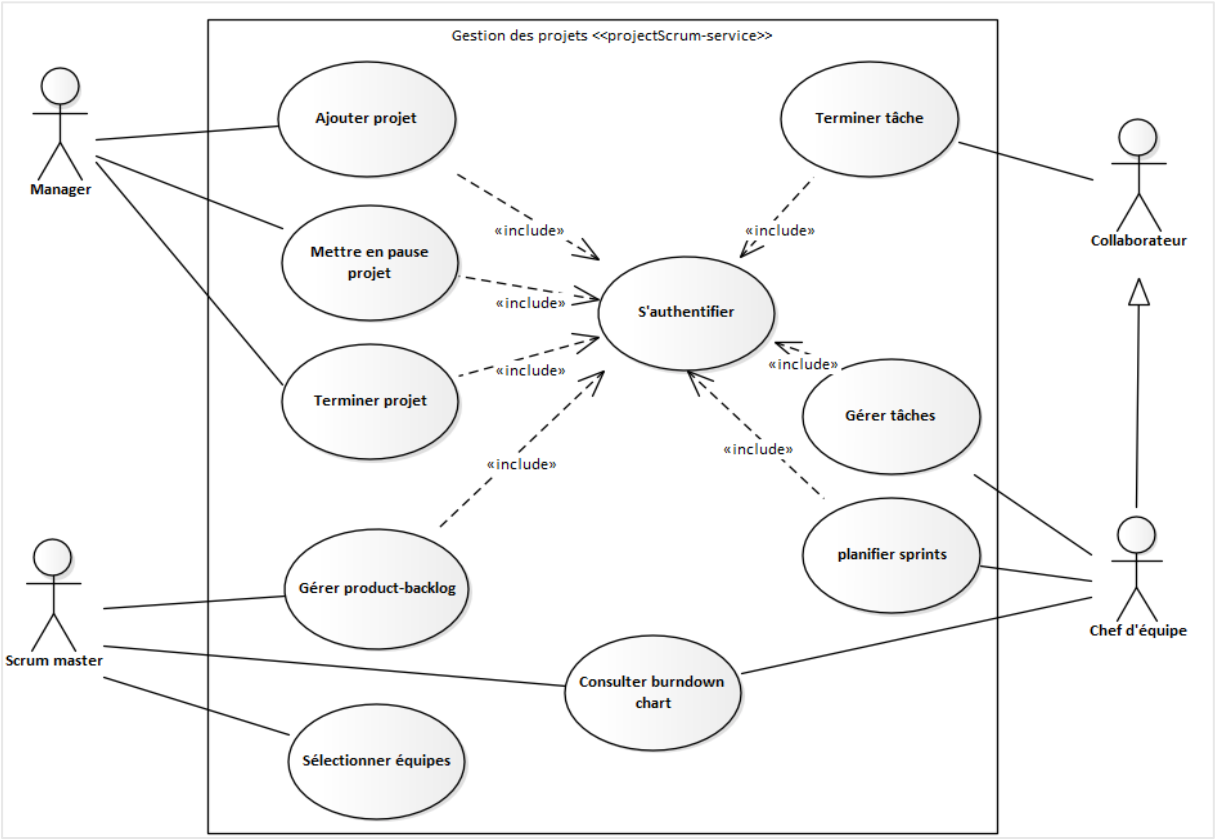


Figure 24: Diagramme de cas d'utilisation <<projectScrum-service-service>>

Pour les cas d'utilisation <<Gérer product-backlog>> et <<Gérer tâches>> sont détaillés par un package correspondant comme indiqué respectivement dans les figures (25) et (26).

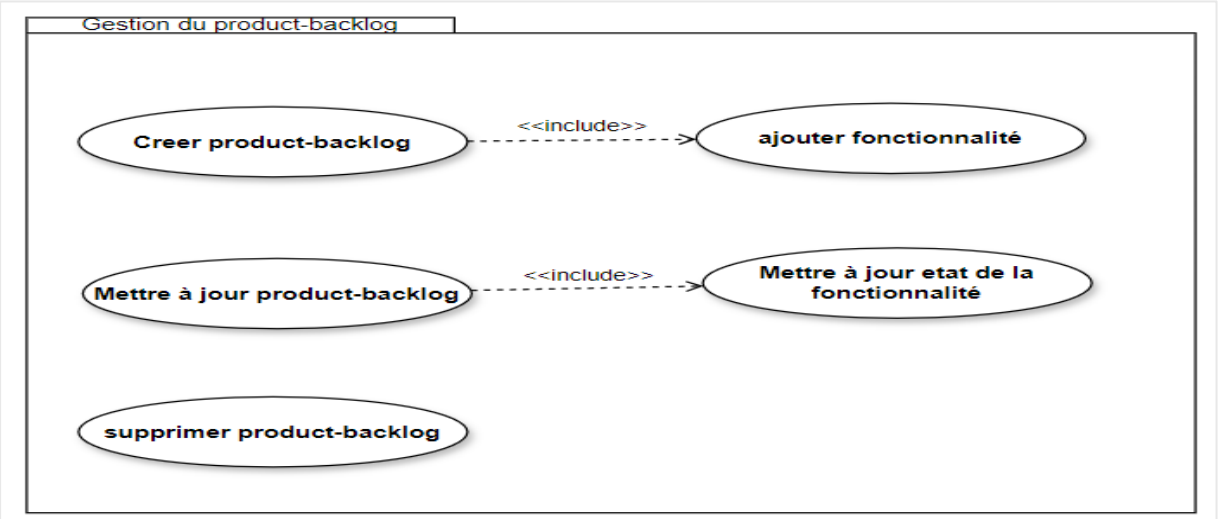


Figure 25 : Paquetage cas d'utilisation <<Gérer product-backlog>>

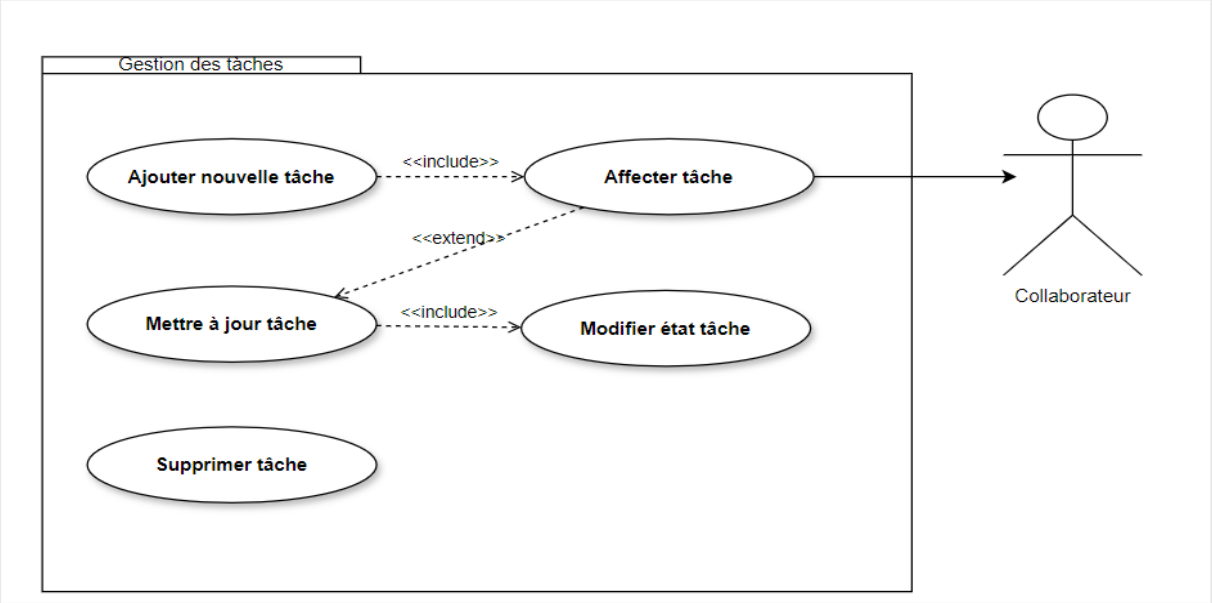


Figure 26: Paquetage cas d'utilisation <<Gérer taches>>

II.4 Module traitement des demandes de congé.

Pour le microservice <<hollidays-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (9).

Cas d'utilisation	*Acteur*	*Description*	*Lien avec autre cas d'utilisation*
Envoyer demande de congé	Tout utilisateur de l'application	Une fois authentifié, un collaborateur peut accéder à l'espace <<Congé>> et s'il dispose d'un solde suffisant, un collaborateur a le droit de	Inclut <<S'authentifier>>

		demander un congé en remplissant un formulaire.	
Annuler demande de congé	Tout utilisateur de l'application	Après avoir envoyé sa demande de congé, et si la demande n'est pas encore traitée par le manager, il peut le collaborateur l'annuler.	Inclut <<S'authentifier>>
Traiter demande de congé	Manager	Une fois authentifié, un collaborateur peut accéder à l'espace <<Congé>>, un manager peut voir les demandes de congé envoyé par les collaborateurs de son département, il peut ainsi l'approuver ou la rejeter.	Inclut <<S'authentifier>>

Tableau 9: Description des cas d'utilisation de <<hollidays-service>>

La figure (27) illustre le diagramme de cas d'utilisation de ce module.

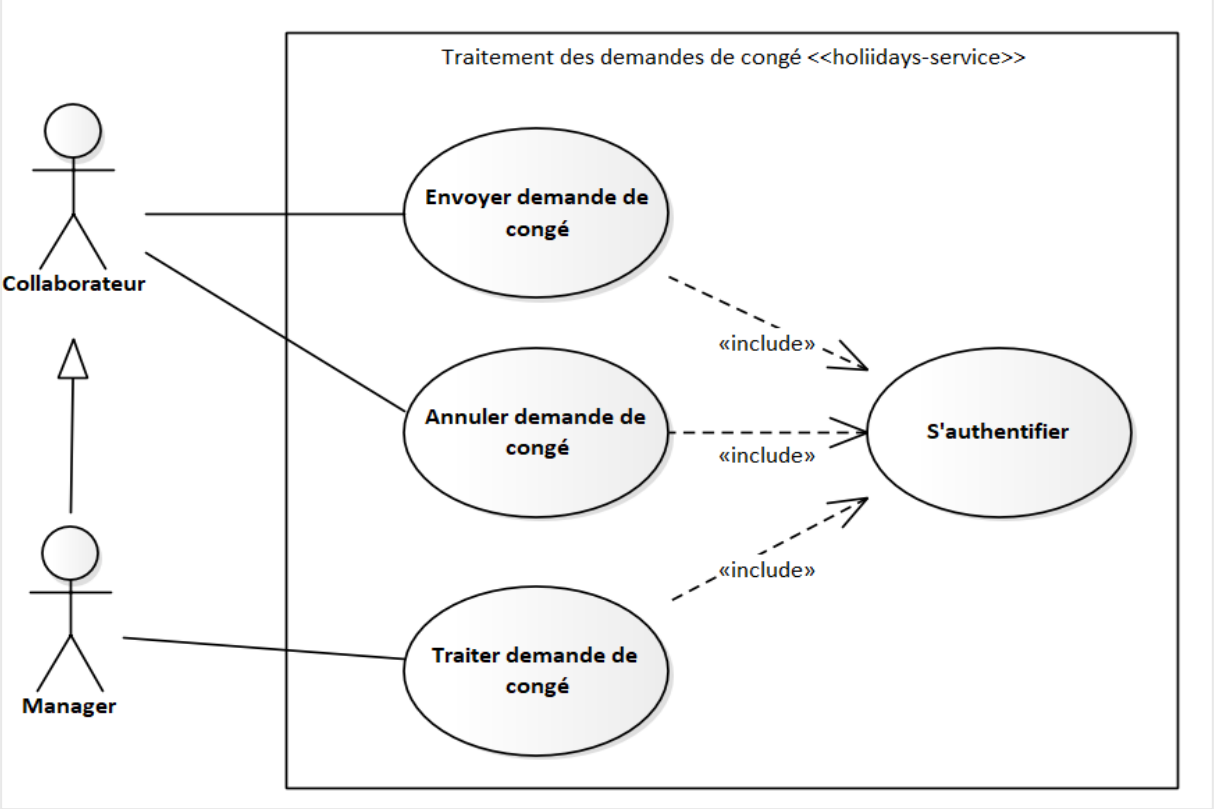


Figure 27: Diagramme de cas d'utilisation <<hollidays-service>>

II.5 Module Feeling board

Pour le microservice <<feelingBoard-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (10).

Cas d'utilisation	Acteur	Description	Lien avec autre cas d'utilisation
Publier sentiment du jour	Tout utilisateur de l'application	Une fois authentifié, un collaborateur peut accéder à l'espace <<Feeling board>> et publier son ressenti du jour en choisissant une des trois options suivantes : satisfait, neutre, insatisfait.	Inclut <<S'authentifier>>
Laisser un feedback	Tout utilisateur de l'application	Après avoir publié son sentiment, un collaborateur peut laisser un commentaire pour son choix.	Inclut <<S'authentifier>>
Consulter sentiment équipe	Chef d'équipe, Manager	Une fois authentifié, un chef d'équipe peut voir le feeling board de son équipe. De même pour le manager.	Inclut <<S'authentifier>>
Consulter sentiment département	Manager	Une fois authentifier, un manager peut consulter le feeling board de toutes les équipes de son département.	Inclut <<S'authentifier>>

Tableau 10: Description des cas d'utilisation de <<feelingBoard-service>>

La figure (28) illustre le diagramme de cas d'utilisation de ce module.

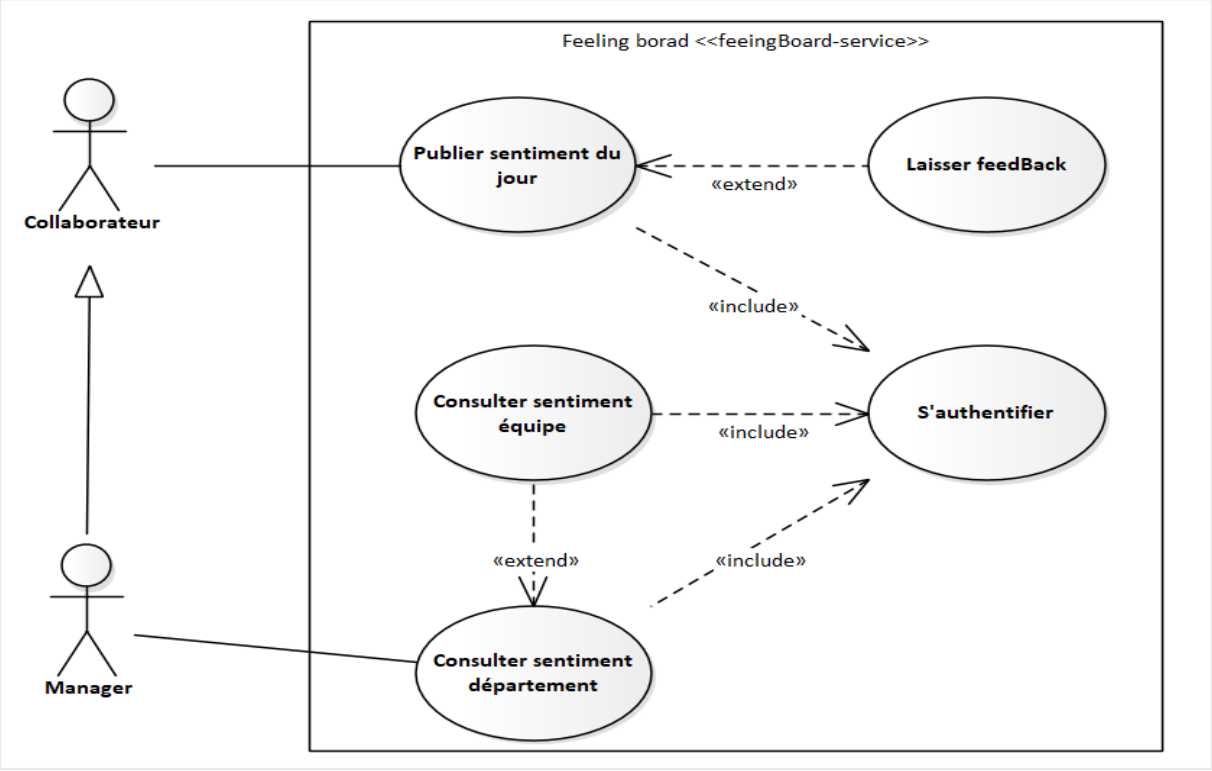


Figure 28: Diagramme de cas d'utilisation <<feelingBoard-service>>

II.6 Module forum de partage

Pour le microservice <<forum-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (11).

Cas d'utilisation	*Acteur*	*Description*	*Lien avec autre cas d'utilisation*
Publier activité	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur peut accéder à l'espace <<Forum>> et publier une activité que ça soit une formation terminée ou un évènement marquant etc..	Inclut <<S'authentifier>>
Modifier activité	Tout utilisateur de l'application.	Après avoir publié son activité dans le forum de partage, un collaborateur peut à tout instant modifier la description ou l'image téléversé dans sa publication.	Inclut <<S'authentifier>>
Supprimer activité	Tout utilisateur de l'application.	Après avoir publié son activité dans le forum de partage, un collaborateur peut à tout instant supprimer sa publication.	Inclut <<S'authentifier>>
Réagir à une activité	Tout utilisateur de l'application.	Une fois accédé à l'espace <<Forum>>, un collaborateur peut réagir à une activité publiée par un de ses collègues dans l'organisation.	Inclut <<S'authentifier>>
Commenter une activité	Tout utilisateur de l'application.	Une fois accédé à l'espace <<Forum>>, un collaborateur peut ajouter un commentaire à une activité publiée par un de ses collègues dans l'organisation.	Inclut <<S'authentifier>>

Tableau 11: Description des cas d'utilisation de <<forum-service>>

La figure (29) illustre le diagramme de cas d'utilisation de ce module.

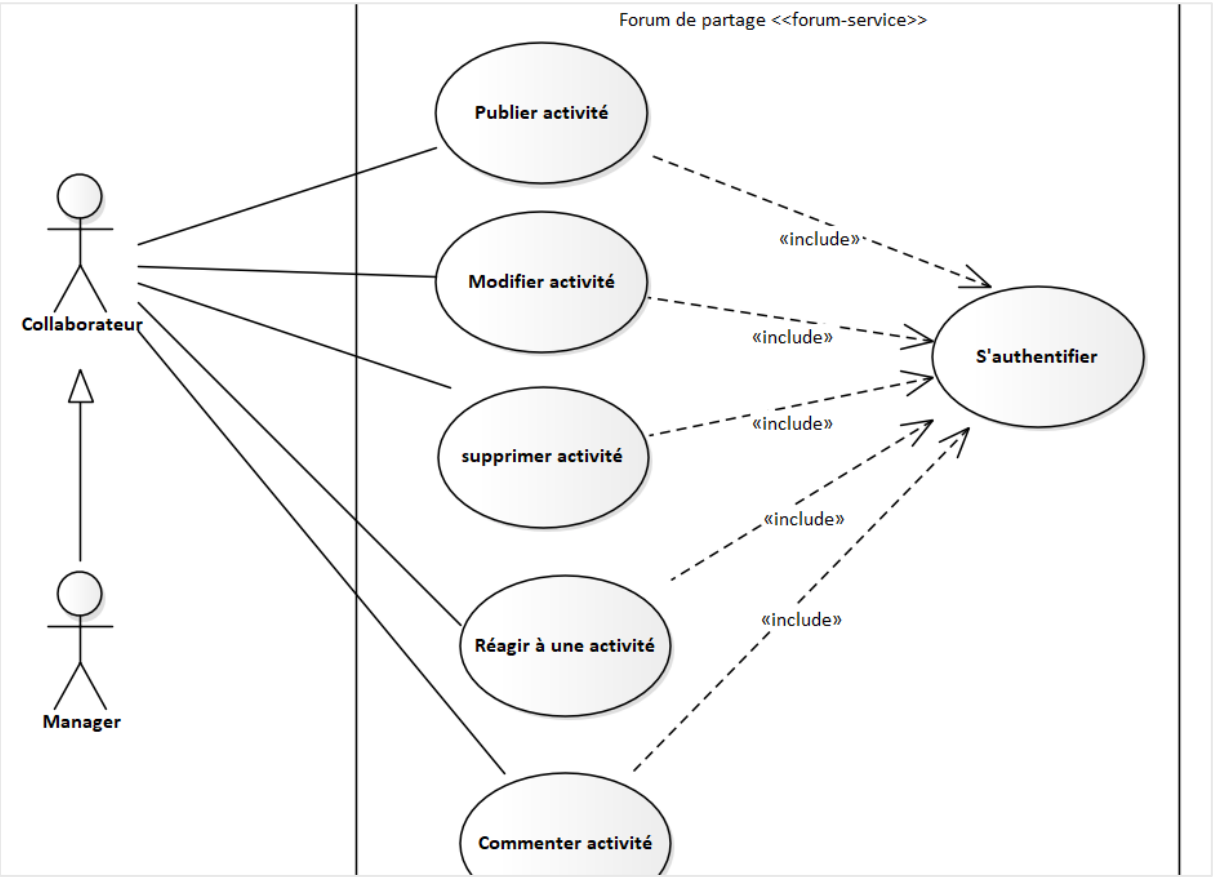


Figure 29: Diagramme de cas d'utilisation <<forum-service>>

II.7 Module chat room

Pour le microservice <<chatRoom-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (12).

Cas d'utilisation	***Acteur***	***Description***	***Lien avec autre cas d'utilisation***
Créer chat room	Chef d'équipe	Une fois authentifié, un collaborateur peut accéder à l'espace <<chat room>> et créer un chat room spécifique à son équipe et ajouter les membres de cette dernière.	Inclut <<S'authentifier>>
Envoyer message	Tout membre du chat Room	Après avoir été ajouté au chat room par son chef d'équipe, un collaborateur membre peut envoyer des messages texte, document etc.	Inclut <<S'authentifier>>
Réagir à un message	Tout membre du chat Room	Un membre du chat room peut réagir aux messages envoyés par ses collègues via des émoticônes.	Inclut <<S'authentifier>>

Chercher un message	Tout membre du chat Room	Un membre du chat room peut à tout instant chercher dans les messages envoyés précédemment dans la conversation.	Inclut <<S'authentifier>>
Filtrer la recherche	Tout membre du chat Room	Quand un collaborateur veut chercher un message dans le chat room, il aura la possibilité de filtrer la recherche en saisissant le teste en question.	Inclut <<S'authentifier>>

Tableau 12: Description des cas d'utilisation de <<chatRoom-service>>

La figure (30) illustre le diagramme de cas d'utilisation de ce module.

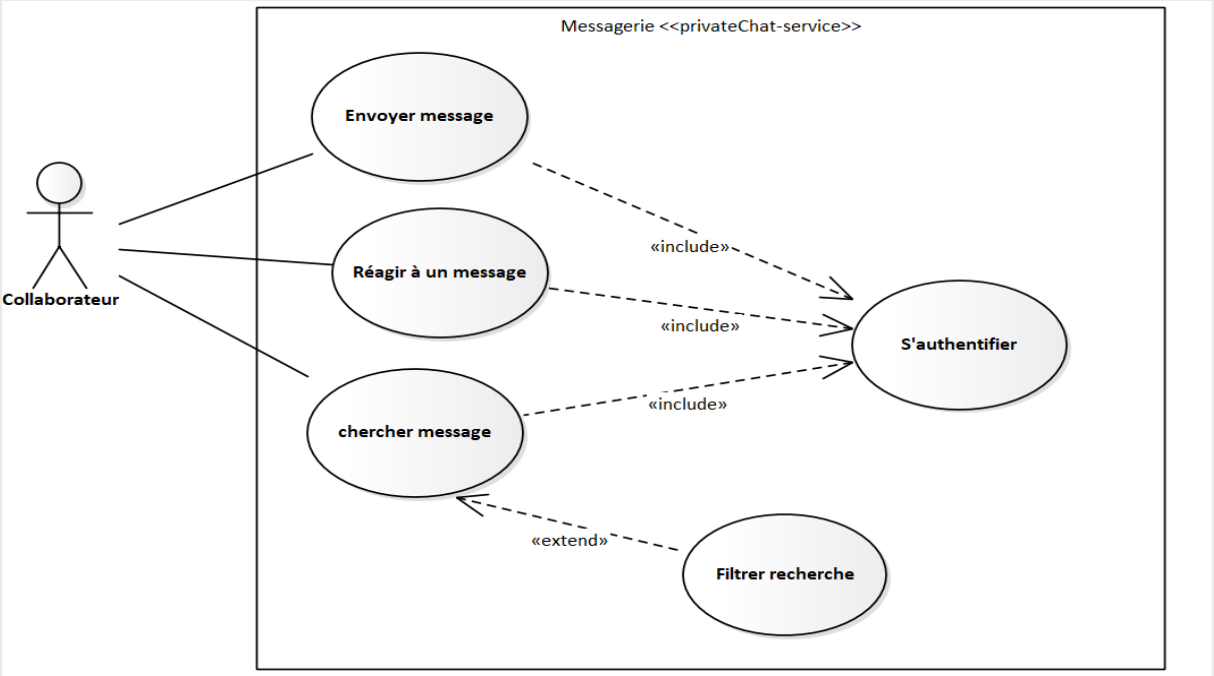


Figure 30: Diagramme de cas d'utilisation <<chatRoom-service>>

II.8 Module messagerie et chat room

Pour le microservice <<privateChat-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (13).

Cas d'utilisation	*Acteur*	*Description*	*Lien avec autre cas d'utilisation*
Envoyer message	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur peut communiquer avec n'importe quel utilisateur de l'application via une conversation privée sécurisée.	Inclut <<S'authentifier>>

Réagir à un message	Tout utilisateur de l'application.	Un collaborateur peut réagir aux messages reçus par ses collègues via des émoticônes.	Inclut <<S'authentifier>>
Chercher un message	Tout utilisateur de l'application.	Un collaborateur peut à tout instant chercher dans les messages envoyés précédemment dans la conversation avec son collègue.	Inclut <<S'authentifier>>
Filtrer la recherche	Tout utilisateur de l'application.	Quand un collaborateur veut chercher un message dans la conversation il aura la possibilité de filtrer la recherche en saisissant le teste en question.	Inclut <<S'authentifier>>

Tableau 13: Description des cas d'utilisation de <<privateChat-service>>

La figure (31) illustre le diagramme de cas d'utilisation de ce module.

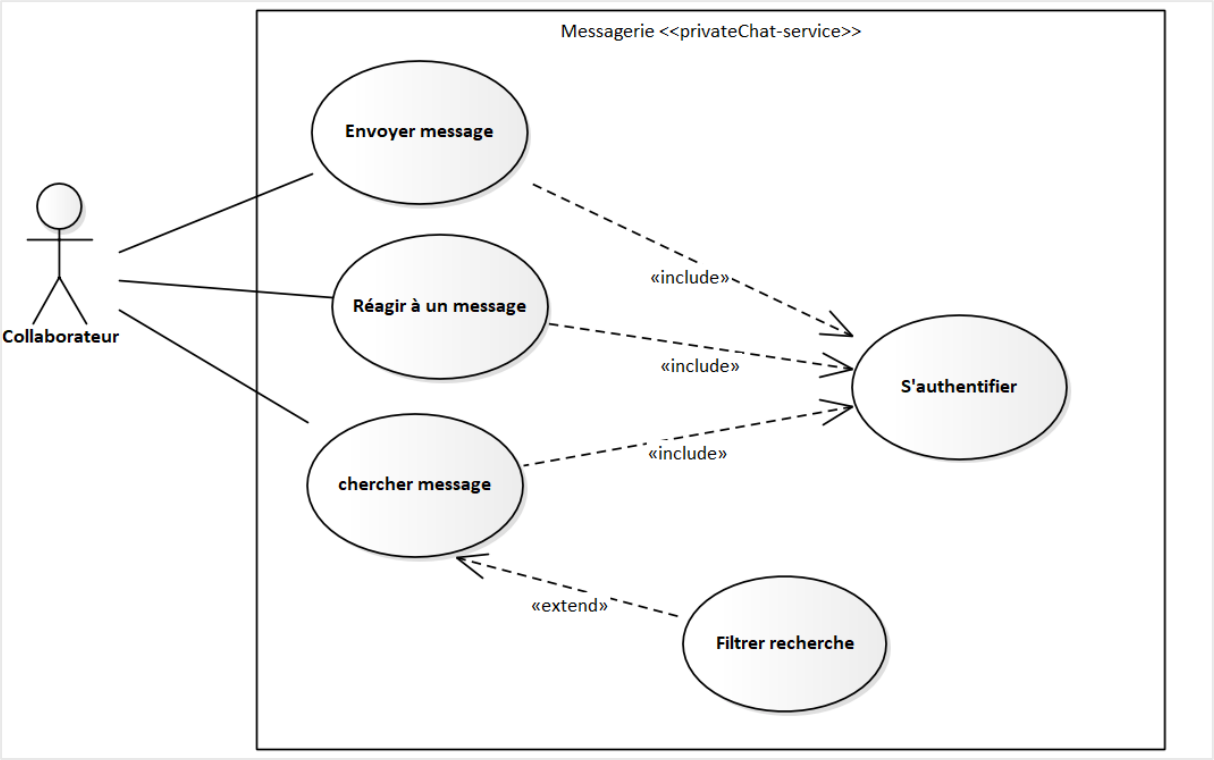


Figure 31: Diagramme de cas d'utilisation <<privateChat-service>>

II.9 Module chatbot RAG

Pour le microservice <<chatbotRAG-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (14).

Cas d'utilisation	*Acteur*	*Description*	*Lien avec autre cas d'utilisation*
Se renseigner sur l'organisation	Tout utilisateur de l'application.	Une fois authentifié, un collaborateur poser des questions aux chatbot pour d'informer sur l'organisation sécurité structure partenariats.	Inclut <<S'authentifier>>
Gérer la base de connaissance	Manager	Le manager peut gérer lz base de connaissance du chatbot en ajoutant de nouvelle connaissance ou supprimant des connaissances existantes etc.	Inclut <<S'authentifier>>

Tableau 14: Description des cas d'utilisation de <<chatbotRAG-service>>

La figure (32) illustre le diagramme de cas d'utilisation de ce module.

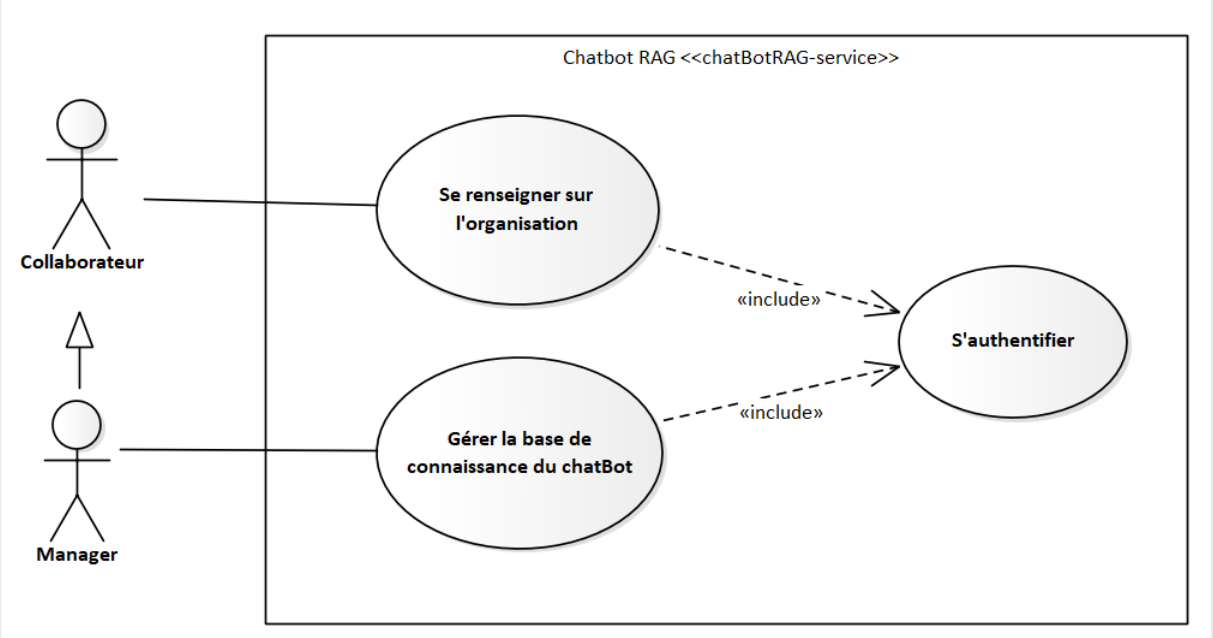


Figure 32: Diagramme de cas d'utilisation <<chatbtRAG-service>>

Pour le cas d'utilisation <<Gérer la base de connaissance du chatbot>> est détaillé via un paquetage comme indiqué dans la figure (33).

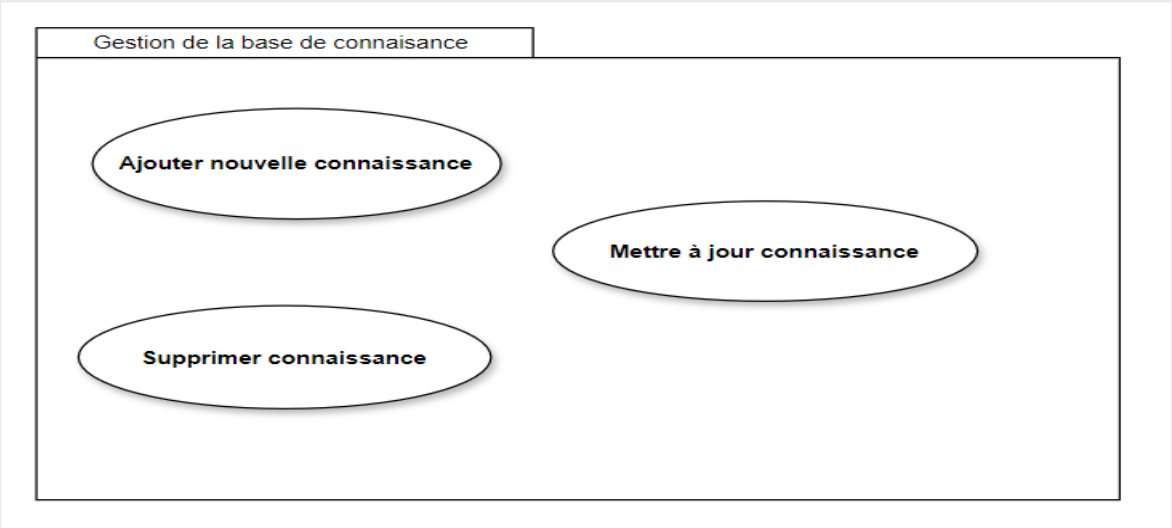


Figure 33: Paquetage cas d'utilisation <<Gérer base de connaissance>>

II.10 Module notification

Pour le microservice <<notification-service>>, nous avons identifié les cas d'utilisation comme illustré dans le tableau (15).

Cas d'utilisation	***Acteur***	***Description***	***Lien avec autre cas d'utilisation***
Consulter notification	Tout utilisateur de l'application.	Une fois une nouvelle notification est reçue par un collaborateur, il peut la consulter pour voir en détails cette nouvelle.	Inclut <<S'authentifier>>
Marquer notification comme lu	Tout utilisateur de l'application.	Une fois une nouvelle notification est reçue par un collaborateur, il peut la marquer comme lu sans la consulter.	Inclut <<S'authentifier>>
Marquer notification comme non lu	Tout utilisateur de l'application.	Un collaborateur peut marquer une notification déjà consulter comme non lu.	Inclut <<S'authentifier>>
Supprimer notification	Tout utilisateur de l'application.	Un collaborateur peut supprimer définitivement une notification, ainsi elle n'est plus affichée.	Inclut <<S'authentifier>>

Tableau 15: Description des cas d'utilisation de <<notification-service>>

La figure (34) illustre le diagramme de cas d'utilisation de ce module.

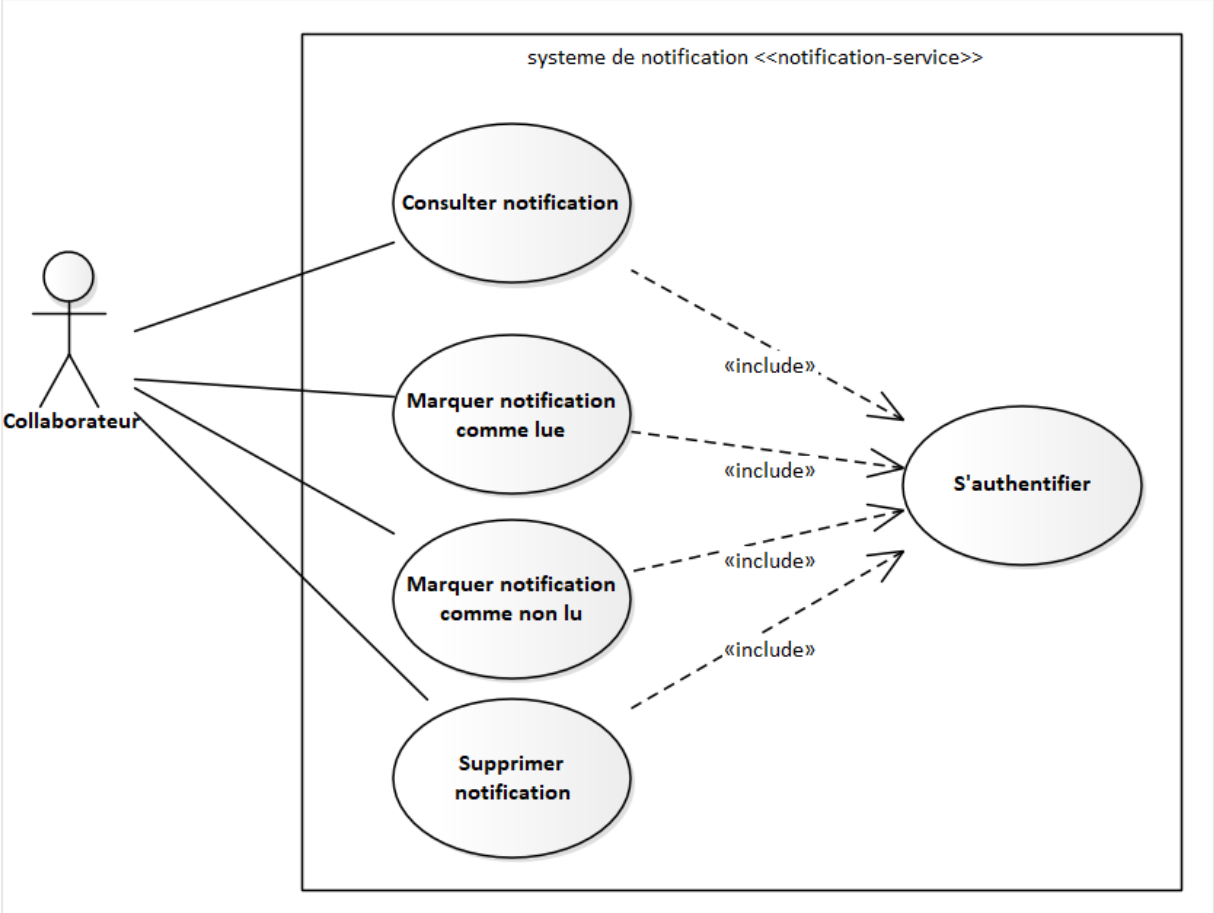


Figure 34: Diagramme de cas d'utilisation <<notification-service>>

III. Synthèse

Au cours de ce chapitre, nous avons détaillé la vue fonctionnelle de notre système, également connue sous le nom de perspective utilisateur. Nous avons décrit les fonctionnalités de chaque module à travers un diagramme d'utilisation correspondant.

La prochaine étape sera consacrée à la vue structurelle de notre projet en fournissant les diagrammes UML convenables.

Chapitre IV : Conception (vue structurelle)

Après avoir réalisé la vue fonctionnelle de notre système à travers les diagrammes d'utilisation, nous allons maintenant élaborer la vue structurelle en utilisant les diagrammes de classes, d'objets et de paquetages.

I. Motivation du choix des diagrammes

Nous avons opté pour l'utilisation des diagrammes **de classes, d'objets et de packages** dans la conception de notre système pour plusieurs raisons. Tout d'abord, ces diagrammes offrent une représentation visuelle claire des différentes classes, de leurs relations, et des regroupements logiques de ces classes en packages, ce qui facilite la compréhension de la structure globale du système.

De plus, ils permettent de modéliser les différents attributs et méthodes de chaque classe, ainsi que les interactions entre les objets, ce qui est essentiel pour une conception précise et détaillée. Les diagrammes de packages ajoutent une couche supplémentaire de clarté en organisant les classes en unités logiques, ce qui simplifie la gestion de la complexité du système.

Leur utilisation nous permettra donc de concevoir efficacement un système robuste et bien structuré, en garantissant une cohérence et une clarté dans la représentation de notre architecture logicielle.

II. Diagramme de classes et son instance

Après avoir détaillé la vue fonctionnelle de notre système, nous allons maintenant élaborer la vue structurelle de nos microservices. Pour ce faire, nous illustrerons chacun d'eux avec un diagramme de classes, accompagné de son instance correspondante, un diagramme d'objets.

II.1 Module Gestion de l'identité.

Pour le microservice <<identity-service>>, nous avons identifié quatre (4) classes à savoir :

➢**User**(id,firstName,lastName,email,mobile,birthday,workStatus,hiringDate,hasProfil Image , lastAuthentication,familyStatus,hollidayBalance).

➢**Profil** (id,profilName).

➢**Team** (id,name,acronym,creationDate,teamLeadEmail).

➢**Department** (id,name,acronym,creationDate,type).

Concernant les énumérations, nous avons trouvé :

➢**ProfilName** (COLLABORATOR,TEAM_LEAD,SCRUM_MASTER,MANAGER).

➢**WorkStatus** (OFFICE,HOLLIDAY,MISSION).

➢**FamilyStatus** (SINGLE,MARRIED,DIVORCED,WIDOWED)

Le diagramme de classes que nous avons élaboré est présenté dans la figure (35).

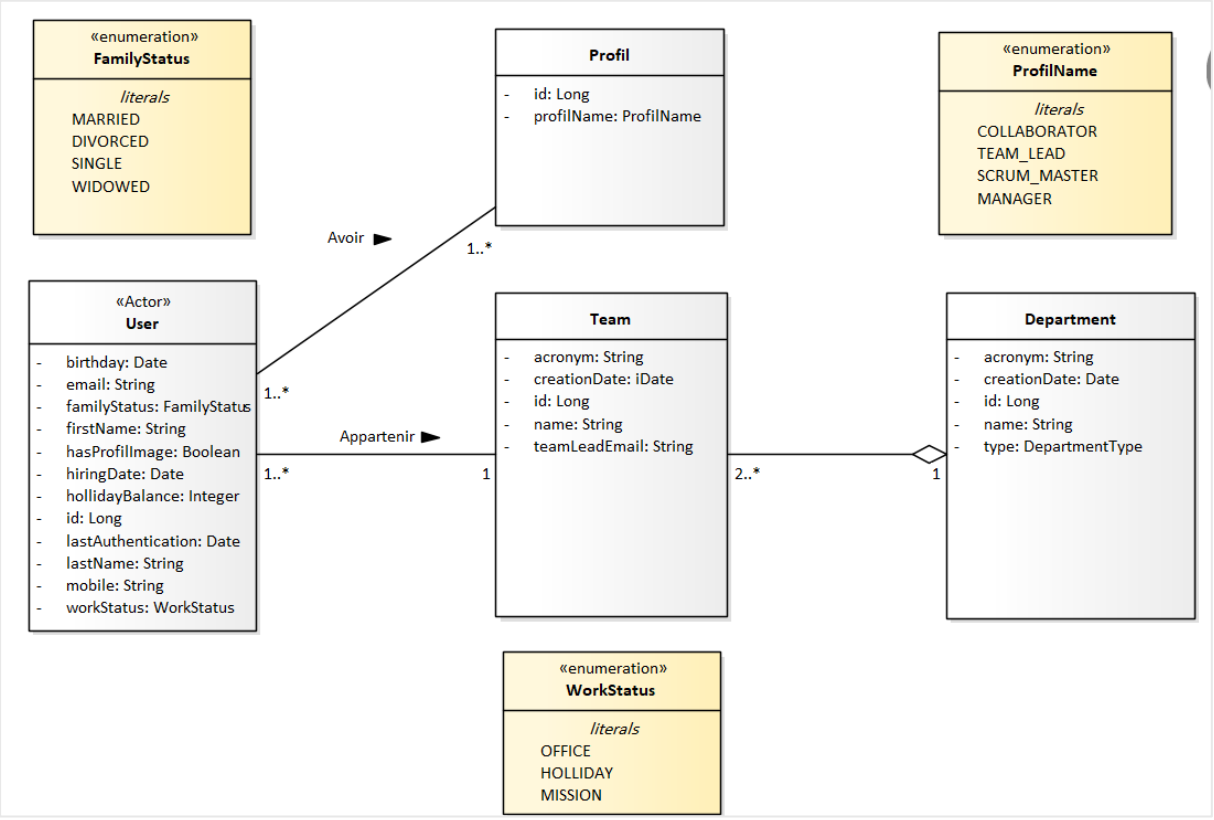


Figure 35: Diagramme de classes <<identity-service>>

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (36).

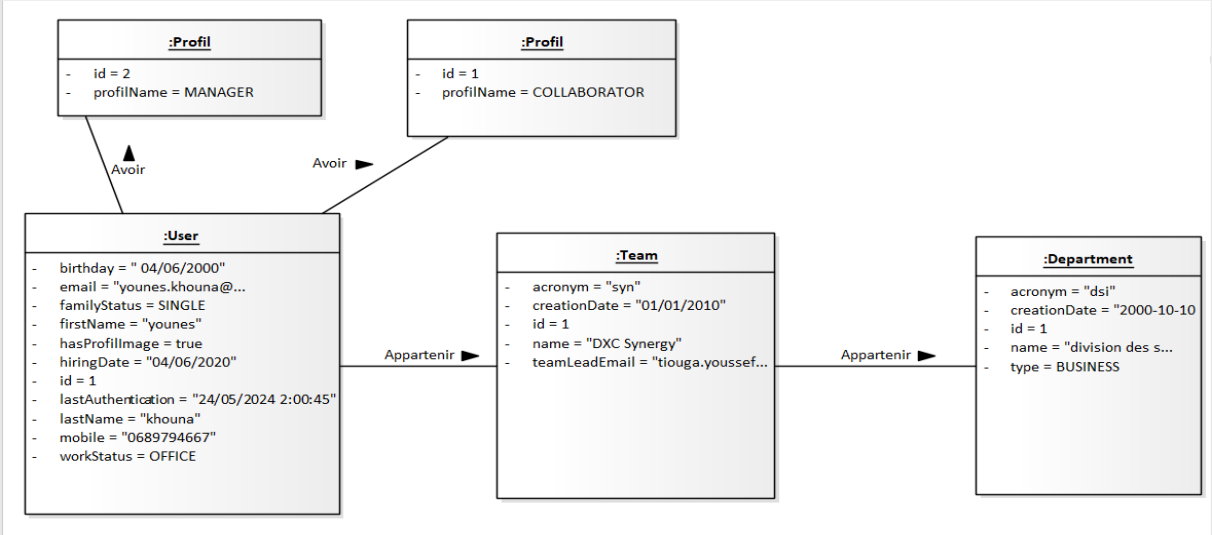


Figure 36: Diagramme d'objets <<identity-service>>

II.2 Module Gestion du patrimoine documentaire

Pour le microservice <<patrimoineDoc-service>>, nous avons identifié deux (2) classes à savoir :

➢**Document**(id,title,description,extension,subject,content,departmentAcronyms, teamAcronyms, emitterEmail).

➢**Version** (id,version,lastUpdate,updatedByEmail).

Concernant les énumérations, nous avons trouvé :

➢**DocumentType**(PRIVATE,TEAM,DEPARTMENT,PUBLIC).

Le diagramme de classes de ce module est présenté dans la figure (37).

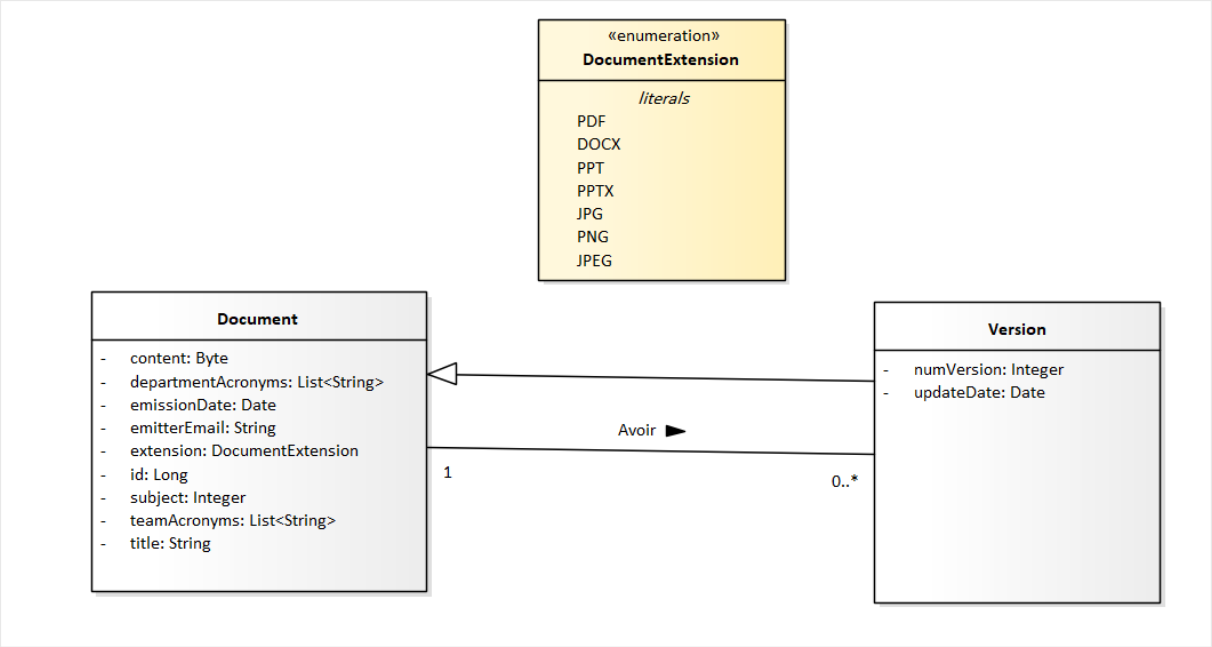


Figure 37: Diagramme de classes <<patrimoineDoc-service>>

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (38).

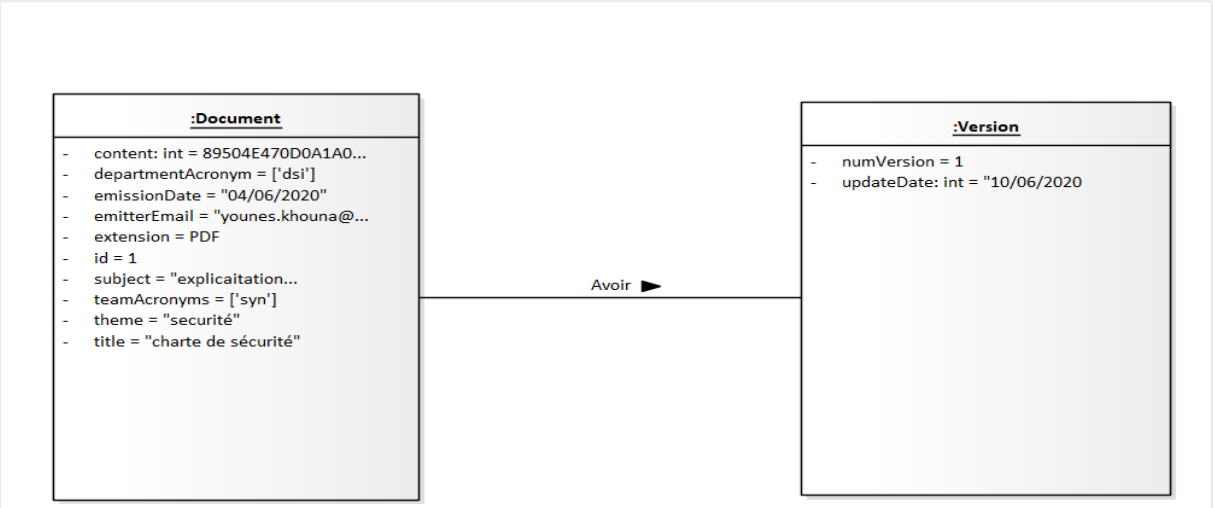


Figure 38: Diagramme d'objets <<patrimoineDoc-service>>

II.3 Module Gestion des projets

Pour le microservice <<projectScrum-service>>, nous avons identifié six (6) classes à savoir :

➢**Project**(id,title,description,debutDate,estimatedEndDate,realEndDate,status,scrumMasterEmail,managerEmail,teamAcronyms,departmentAcronyms).

➢**ProductBacklog**(id,title,creationDate,lastUpdateDate).

➢**Functionality**(id,title,description,priority,complexity,state).

➢**Sprint**(id,title,description,debutDate,endDate,state).

➢**Task(**id,title,description,debutDate,estimatedDate,realEndDate,status,userEmail).

➢**Increment** (id,title,description,efficiency).

Concernant les énumérations, nous avons trouvé :

➢**ProjectStatus**(TO_DO,IN_PROGRESS,FINISHED,CANCELED).

➢**FunctionalityState**(TO_DO,IMPLEMENTED).

➢**FunctionalityComplexity**(HARD,MODERATE,EASY).

➢**FunctionalityPriority**(HIGH,MEDIUM,LOW).

➢**SprintState**(FINISHED,IN_PROGRESS).

➢**TaskStatus**(TO_DO,IN_PROGRESS,FINISHED).

Le diagramme de classes que nous avons élaboré est présenté dans la figure (39).

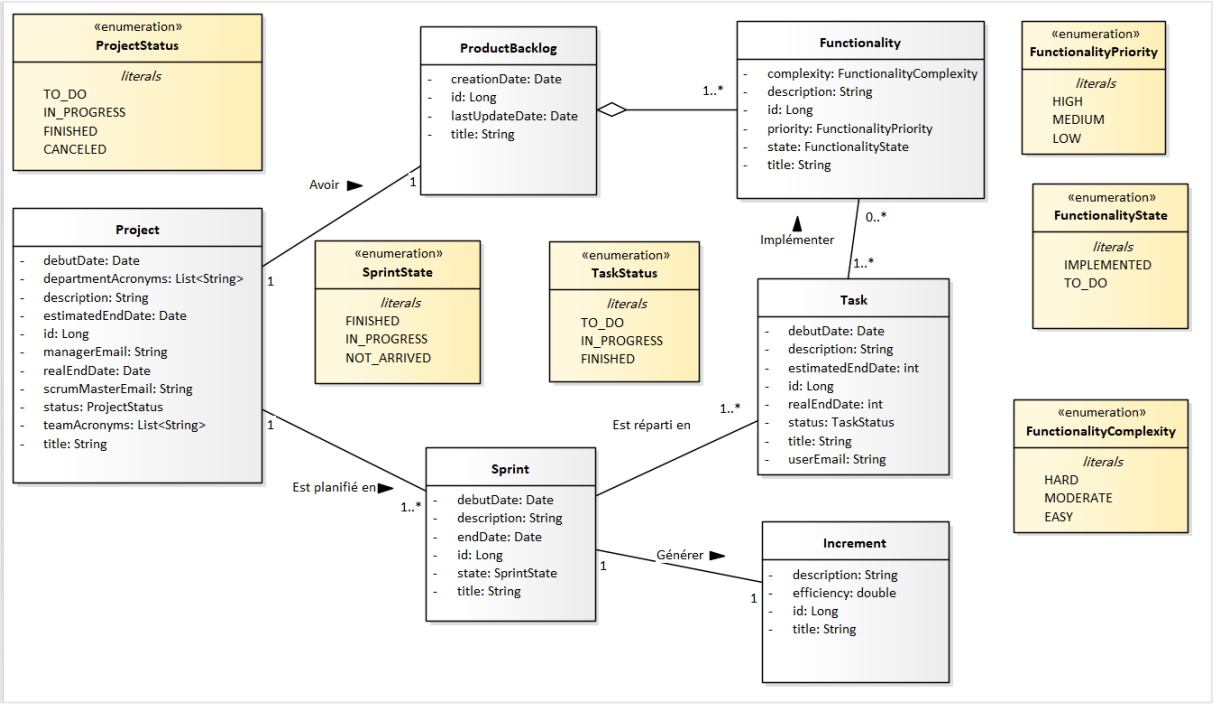


Figure 39: Diagramme de classes <<projectScrum-service>>

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (40).

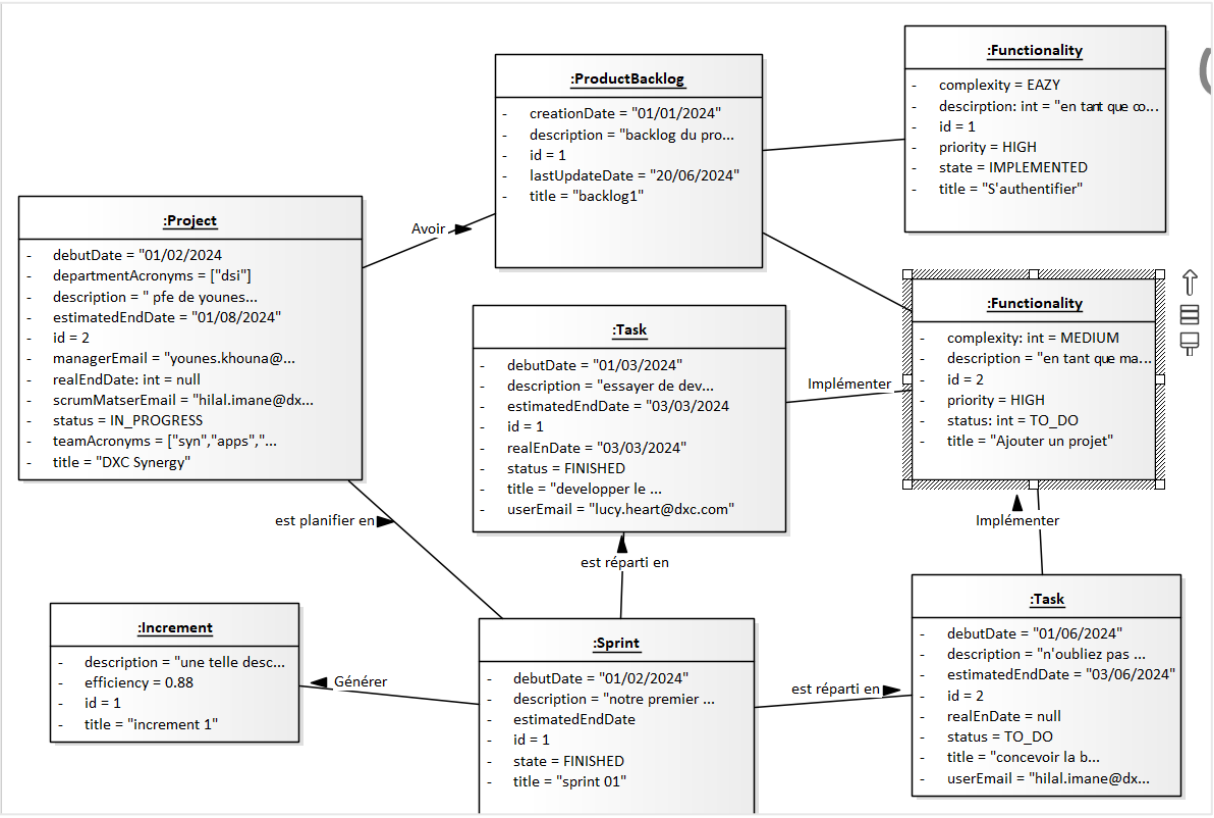


Figure 40: Diagramme d'objets <<projectScrum-service>>

II.4 Module traitement des demandes de congé

Pour le microservice <<hollidays-service>>, nous avons identifié une seule (1) classe à savoir :

➢**HollidayRequest**(id,debutDate,endDate,requestDate,decisonDate,status,type,backupEmail,managerEamil,userEmail,currentHollidayBalance).

Concernant les énumérations, nous avons trouvé :

➢**HollidayStatus**(IN_PROGRESS ,CANCELED,APPROVED,REJECTED).

➢**HollidayType(**NORMAL,WEDDING,MATERNITY,PATERNITY,SICKNESS,DEATH).

Le diagramme de classes que nous avons élaboré est présenté dans la figure (41).

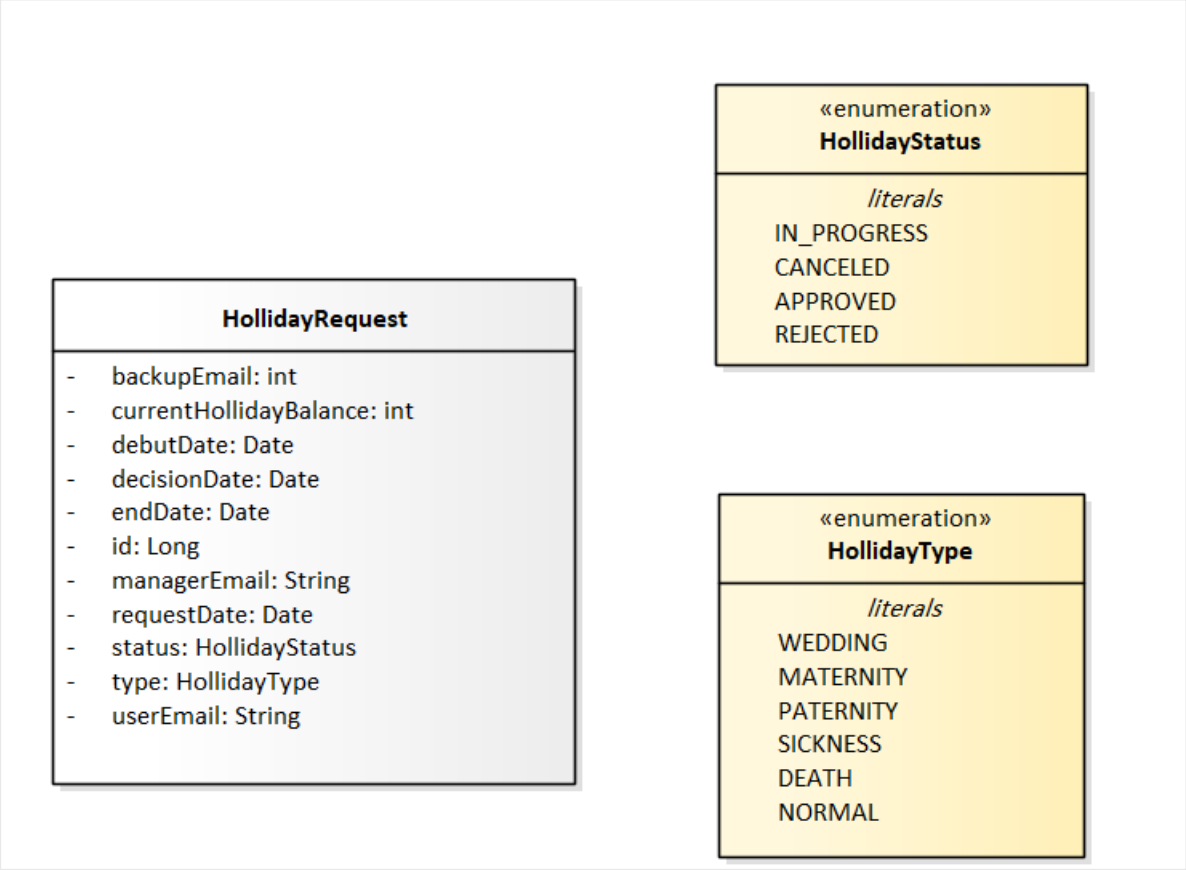


Figure 41: Diagramme de classes <<hollidays-service>>

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (42).

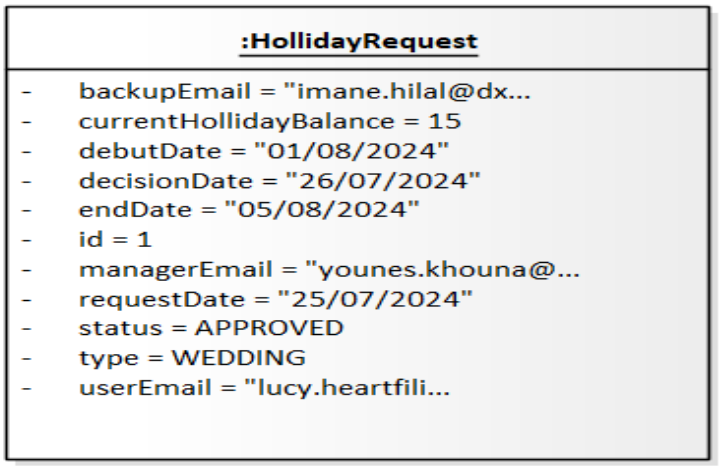


Figure 42: Diagramme d'objets <<hollidays-service>>

II.5 Module feeling board

Pour le microservice <<feelingBoard-service>>, nous avons identifié une seule (1) classe à savoir :

➢**Feeling**(id,reference,feelingStatus,opinion,sentDate,userEmail)

Concernant les énumérations, nous avons trouvé :

➢**FeelingStatus**(SATISFIED,NEUTRAL,UNSATISFIED).

Le diagramme de classes que nous avons élaboré est présenté dans la figure (43).

Figure 43: Diagramme de classes <<feelingBoard-service>>

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (44).

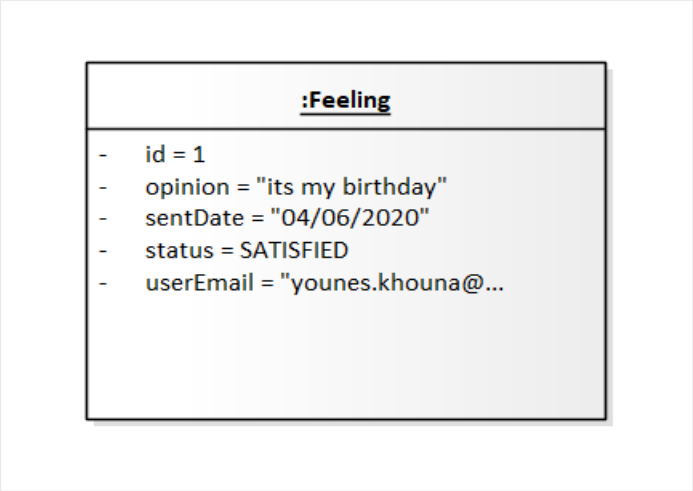


Figure 44: Diagramme d'objets <<feelingBoard-service>>

II.6 Module forum de partage

Pour le microservice <<forum-service>>, nous avons identifié trois (3) classes à savoir :

➢**POST**(id,description,image,postDate,userEmail)

➢**Reaction(**(id,reactionDate,type,userEmail)

➢**Comment**(id,content,sentDate,userEmail)

Concernant les énumérations, nous avons trouvé :

➢**ReactionType** (LIKE,LAUGH,SUPPORT,INFORMATIVE,CONGRATS).

Le diagramme de classes que nous avons élaboré est présenté dans la figure (46).

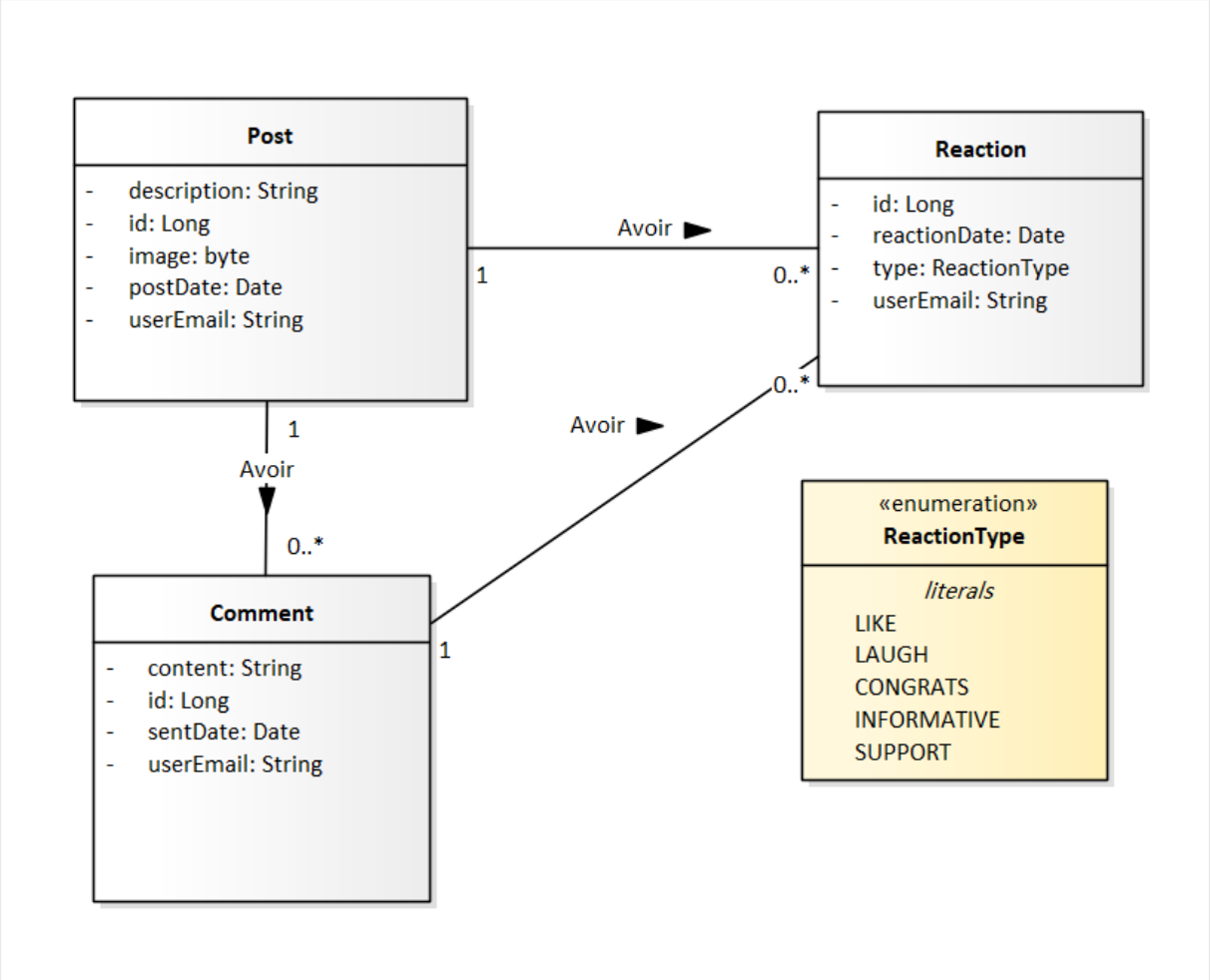


Figure 45: Diagramme de classes « forum-service »

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (46).

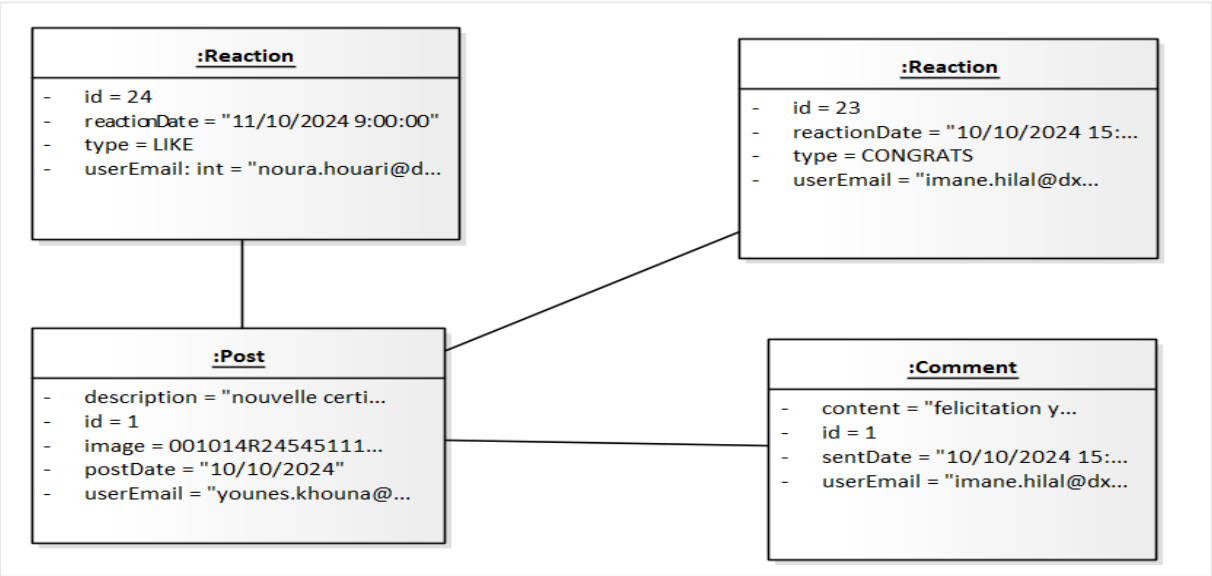


Figure 46: Diagramme d'objets <<forum-service>>

II.7 Module chat room

Pour le microservice <<chatRoom-service>>, nous avons identifié trois (3) classes à savoir :

➢**ChatRoom**(id,description,name,creationDate, teamAcronym)

➢**Reaction(**(id,reactionDate,type,userEmail)

➢**Comment**(id,content,sentDate,userEmail)

Concernant les énumérations, nous avons trouvé :

➢**ReactionType** (LIKE,LAUGH,SUPPORT,INFORMATIVE,CONGRATS).

Le diagramme de classes que nous avons élaboré est présenté dans la figure (47).

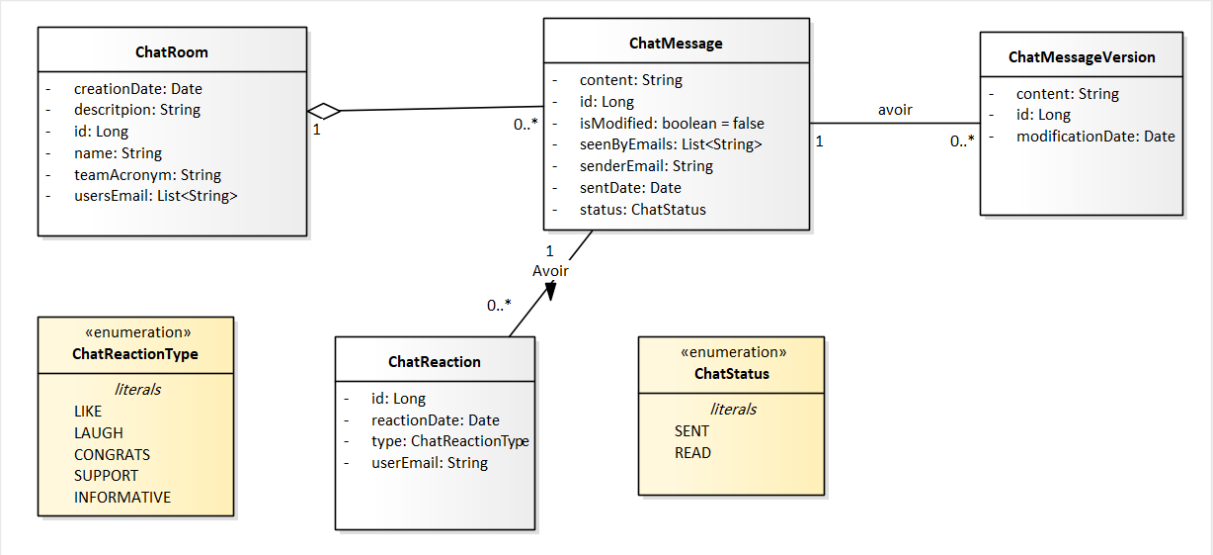


Figure 47: Diagramme de classes <<chatRoom-service>>

Ainsi, le diagramme d'objet correspondant est illustré dans la figure (48).

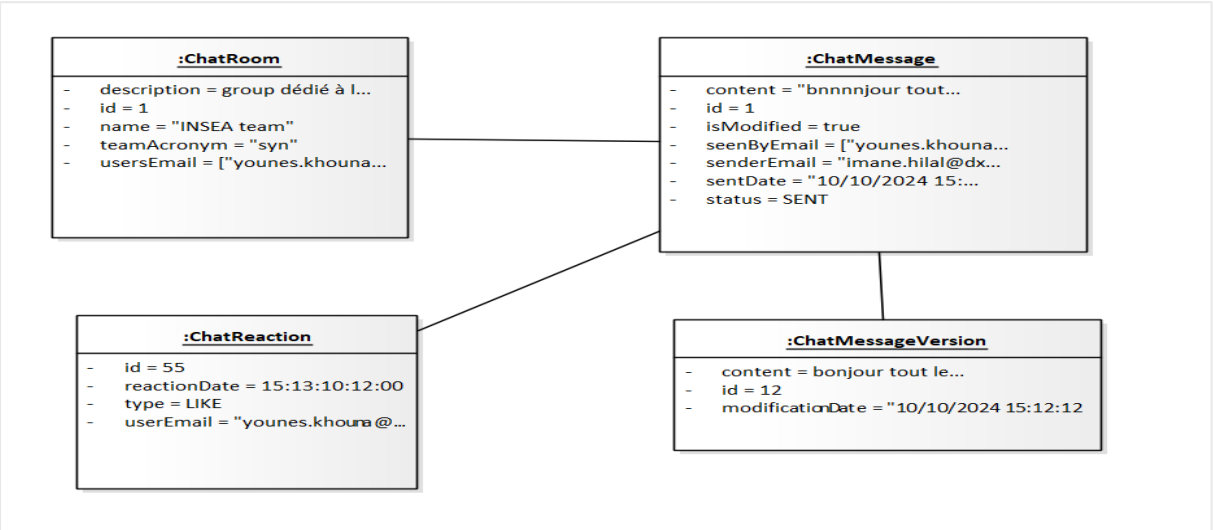


Figure 48: Diagramme d'objets <<chatRoom-service>>

II.8 Module messagerie

Pour le microservice <<privateChat-service>>, la même approche de modélisation que celle utilisée pour le module précédent, à savoir le microservice <<chatRoom-service>>, a été appliquée.

II.9 Module chatbot RAG

Pour le microservice <<chatBotRag-service>>, nous n'avons pas conçu de base de données spécifique, car nous consommerons directement les LLMs fournis par le framework Spring AI. L'architecture de ce module sera détaillée dans le chapitre III (Architecture de l'application).

II.10 Module notification

Pour le microservice <<notification-service>>, nous avons identifié une seule (1) classe à savoir :

➢**Notification** (id,content,reference,sentDate,status,type,userEmail).

Concernant les énumérations, nous avons trouvé :

➢**NotificationStatus** (READ,NOT_READ).

➢**NotificationType** (PRIVATE,TEAM,DEPARTMENT,PUBLIC).

Le diagramme de classes que nous avons élaboré est présenté dans la figure (49).

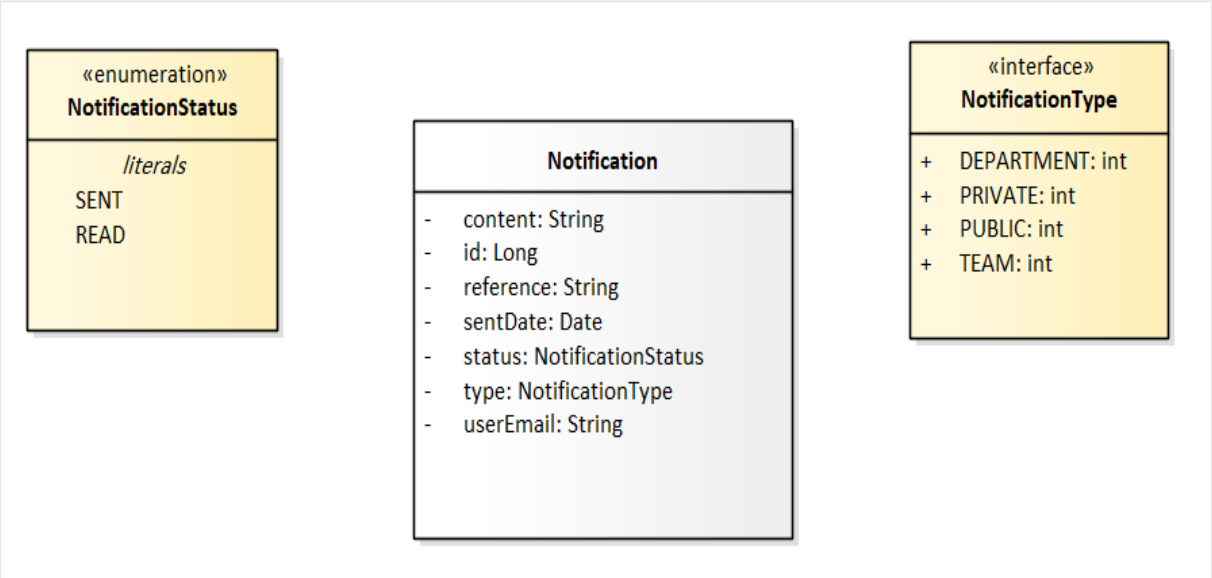


Figure 49: Diagramme de classes <<notification-service>>

III. Diagramme de paquetages

Nous avons utilisé ce type de diagramme pour modéliser de façon plus claire la structure de l'organisation de notre code,

III.1 Diagramme de paquetage global

Chaque microservices de l'application représente un package indépendant ce qui explique **l'absence de dépendance** entre les différents paquetages de niveau (1).

La figure (50) illustre le diagramme de paquetage global de l'application.

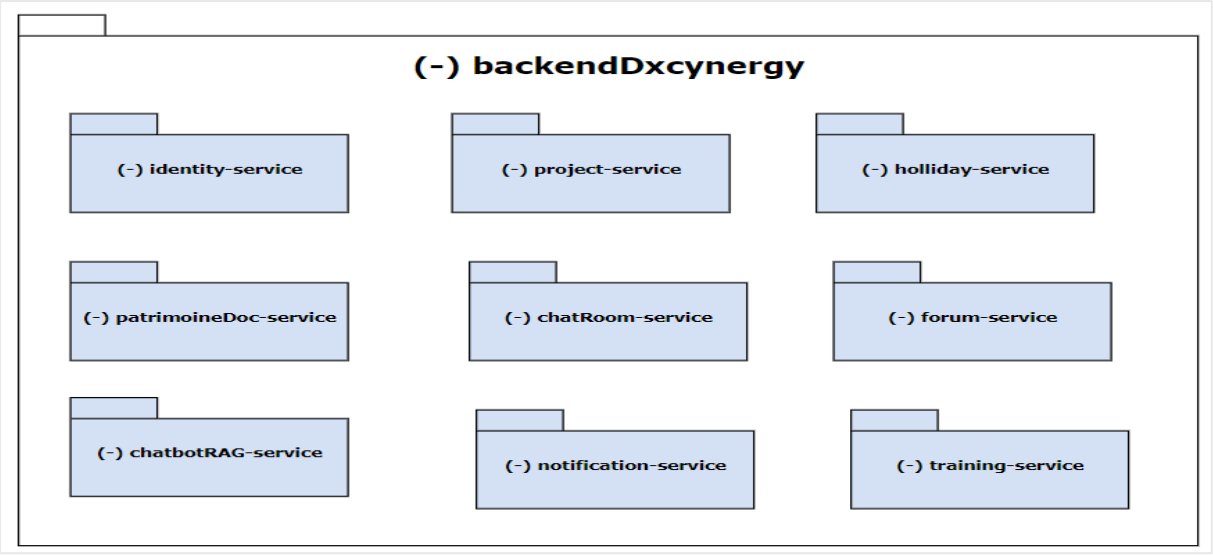


Figure 50: Diagramme de paquetages global

III.2 Etude de cas : Diagramme de paquetage de « identity-service>>

Nous avons détaillé la structure du code du micro-service « identity-service » par un diagramme de paquetages regroupant tous les paquetages fils de ce dernier.

Les figures (51) et (52) illustrent ce diagramme.

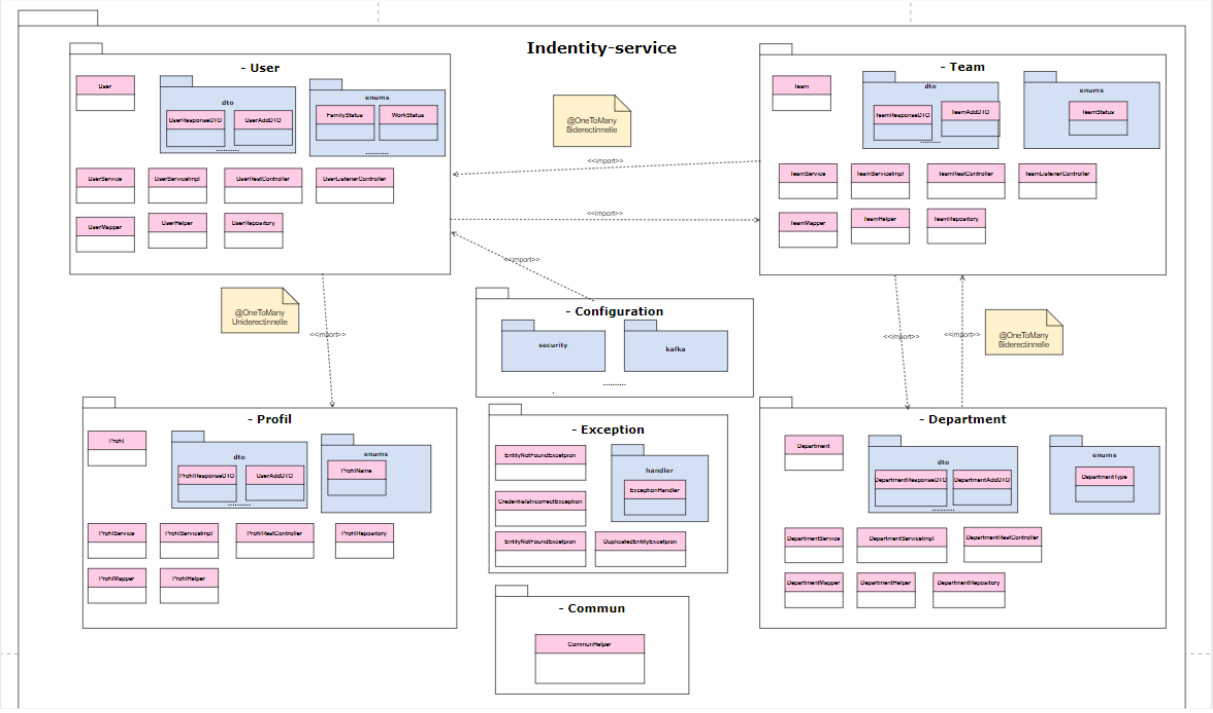


Figure 51: Diagramme de paquetages "identity-service"

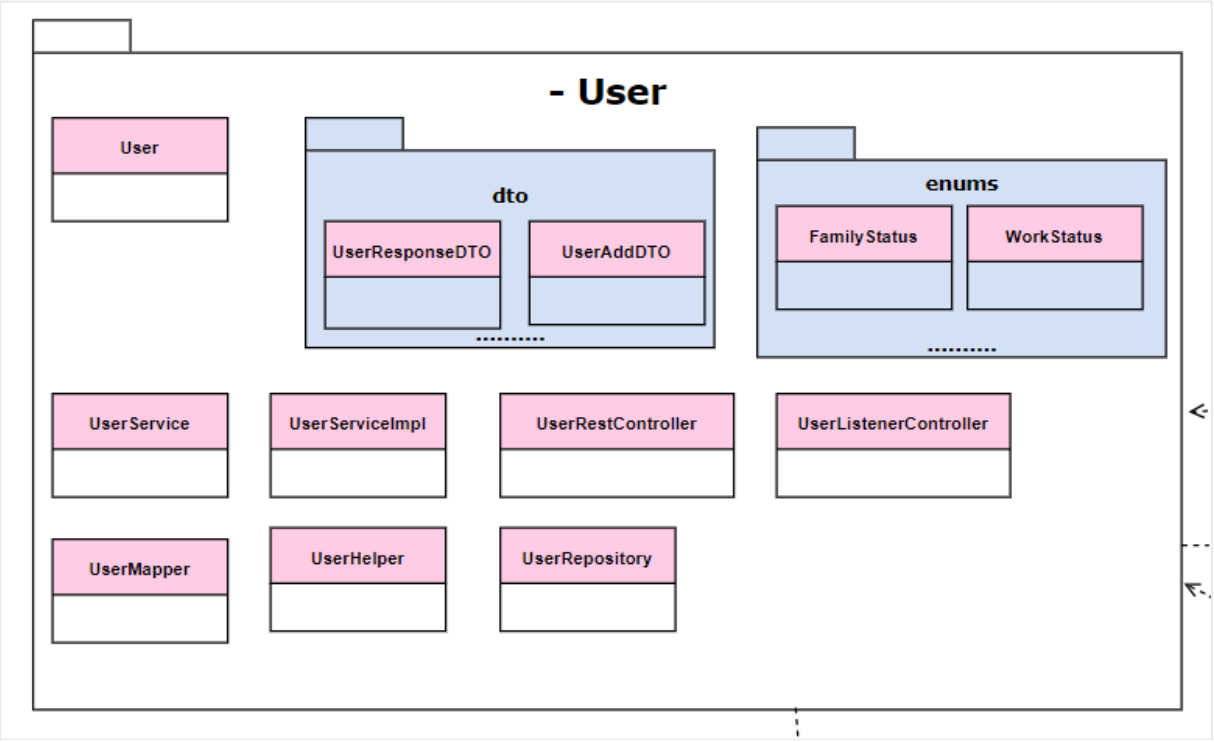


Figure 52: Diagramme de paquetages "User"

IV. Synthèse

Au cours de ce chapitre, nous avons établi la vue structurelle de notre système en utilisant les diagrammes de classes, d'objets et de paquetages. Ces outils nous ont permis de modéliser précisément la structure interne du système, en identifiant les classes, les attributs, les méthodes et les relations entre les objets.

La prochaine étape, nous aborderons maintenant la vue dynamique, qui met l'accent sur le comportement et les interactions en temps réel entre les différents composants. Cette perspective nous permettra de comprendre comment le système réagit et évolue dans des scénarios d'utilisation variés.

Chapitre V : Conception (vue dynamique)

Après avoir terminé la vue fonctionnelle et structurelle de notre système, nous allons maintenant clarifier la vue dynamique en élaborant les diagrammes de séquence et d'état-transition pour quelques cas d'utilisation.

Dans cette partie, tous les diagrammes de séquence que nous avons élaborés sont modélisés selon l'approche MVC, ce qui offre une vue transversale et claire des scénarios de nos cas d'utilisation.

I. Motivation du choix des diagrammes

Nous avons également choisi d'intégrer les diagrammes **de séquence et d'état-transition** dans la conception de notre système pour plusieurs raisons. Tout d'abord, cette combinaison optimale offre une représentation visuelle dynamique des interactions entre les différents composants du système.

Les diagrammes de séquence permettent de modéliser les séquences d'actions entre les objets, offrant ainsi une vue détaillée du flux d'informations et des communications entre les éléments du système. D'autre part, les diagrammes d'état-transition mettent en évidence les différents états que peuvent traverser les objets et les transitions entre ces états en réponse à des événements spécifiques, ce qui permet de modéliser le comportement dynamique du système.

Cette approche nous permettra de mieux comprendre et de visualiser les interactions en temps réel entre les composants du système, contribuant ainsi à une conception précise et bien structurée.

II. Diagrammes de séquences

Contrairement à la vue fonctionnelle et structurelle, dans cette partie, nous allons introduire le facteur temporel dans l'interaction entre le système et l'utilisateur final. Nous décrirons les principaux cas d'utilisation à travers un diagramme de séquence pour chacun d'eux.

II.1 Diagramme de séquences du cas d'utilisation : S'authentifier

Nom du cas d'utilisation	S'authentifier
Acteur	Collaborateur
Précondition	Aucune
Scénario nominal	-Accéder à la page d'authentification. -Saisir ses identifiants (email et mot de passe). -Cliquer sur le bouton « se connecter ». -Vérifier la validité des champs remplis. -Vérifier l'existence de l'utilisateur dans la base de données. -Stocker le jeton de l'utilisateur dans la mémoire du navigateur de ce dernier. -Créer la session utilisateur et diriger vers espace accueil.
Scénario d'échec (1)	-Saisir ses identifiants (email et mot de passe). -Cliquer sur le bouton « se connecter ».

	-Invalidité des champs saisis.
Scénario d'échec (2)	-Saisir ses identifiants (email et mot de passe). -Cliquer sur le bouton « se connecter ». -Vérifier la validité des champs remplis. -L'utilisateur n'existe pas dans la base de données.
Postcondition	Accéder à l'espace « Accueil » de la plateforme.

Tableau 16: Description des scénarios "S'authentifier"

La figure (53) illustre le diagramme de séquences du cas d'utilisation « S'authentifier ».

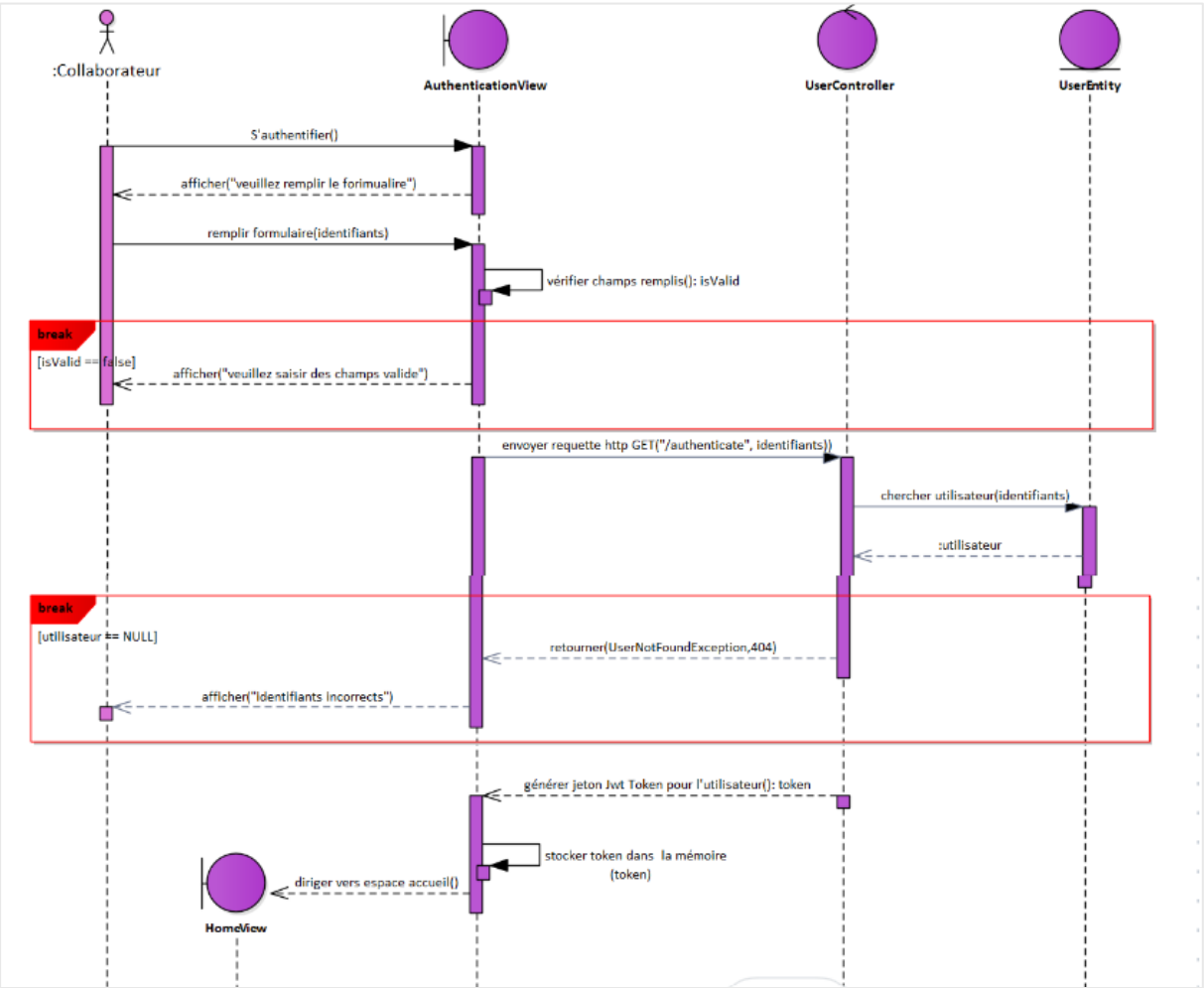


Figure 53: Diagramme de séquences <<S'authentifier>>

II.2 Diagramme de séquences du cas d'utilisation : Ajouter une équipe

Nom du cas d'utilisation	Ajouter une équipe
Acteur	Manager
Précondition	Authentification
Scénario nominal	-Accéder à l'espace « mon département ». -Remplir le formulaire d'ajout de l'équipe (nom, acronyme, dateCreation, chefEquipe, membres).

	-Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. -Vérifier l'existence d'une autre équipe avec le même acronyme. -Enregistrer la nouvelle équipe dans la base de données. -Afficher un message de réussite de l'opération.
Scénario d'échec (1)	-Remplir le formulaire d'ajout de l'équipe (nom ,acronyme , dateCreation, chefEquipe , membres). -Cliquer sur le bouton « Ajouter ». **-Invalidité des champs saisis.**
Scénario d'échec (2)	-Remplir le formulaire d'ajout de l'équipe (nom, acronyme, dateCreation, chefEquipe, membres). -Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. **- Une autre équipe avec le même nom existe déjà dans la base de données.**
Postcondition	Equipe ajoutée avec succès.

Tableau 17: Description des scénarios "Ajouter une équiper"

La figure (54) illustre le diagramme de séquences du cas d'utilisation « Ajouter une équipe ».

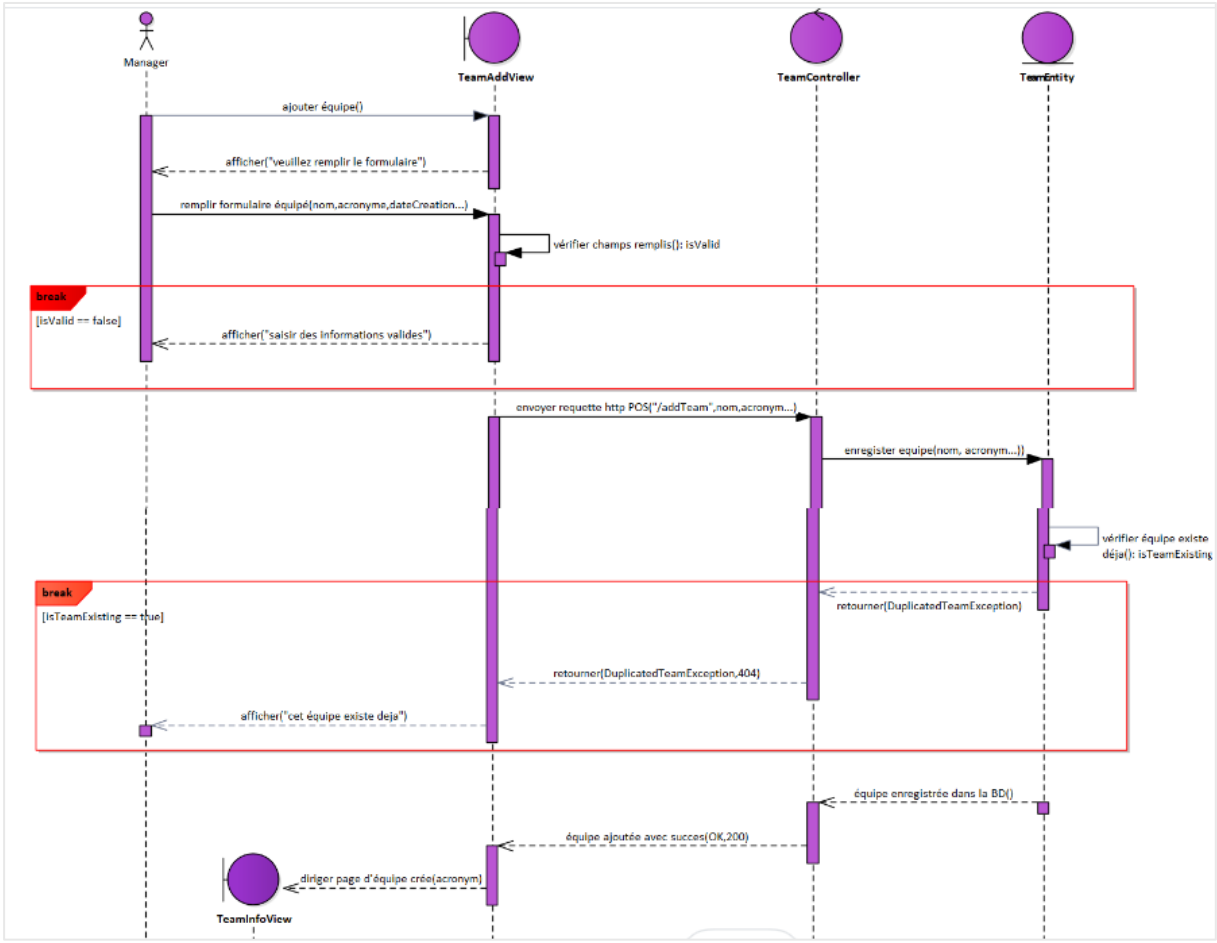


Figure 54: Diagramme de séquences <<Ajouter une équiper>>

II.3 Diagramme de séquences du cas d'utilisation : Consulter un document

Nom du cas d'utilisation	Consulter document
Acteur	Collaborateur
Précondition	Authentification
Scénario nominal	-Accéder à l'espace « Documents ». -Cliquer sur le document à consulter. - Récupérer le contenu du document depuis la base de données sous format Byte. -Décompresser le contenue. -Afficher le contenue du document.
Scénario d'échec	-Cliquer sur le document à consulter -Récupérer le contenu du document depuis la base de données sous format Byte. -Erreur de décompression.
Postcondition	Affichage la page du document avec son contenu et informations de bases(titre,date,auteur…)

Tableau 18: Description des scénarios "Consulter un document"

La figure (55) illustre le diagramme de séquences du cas d'utilisation « Consulter document »

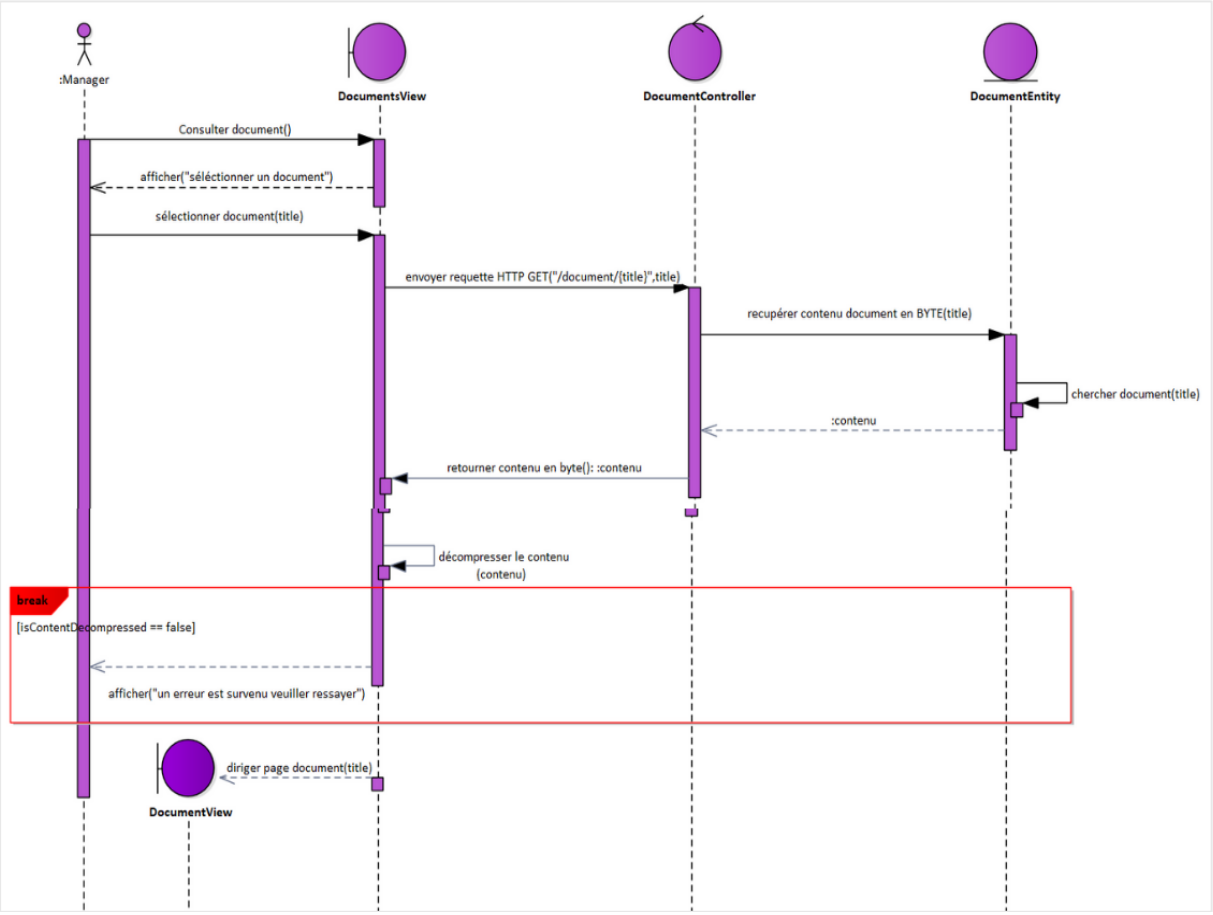


Figure 55: Diagramme de séquences <<Consulter document>>

II.4 Diagramme de séquences du cas d'utilisation : Ajouter un projet

Nom du cas d'utilisation	Ajouter un projet
Acteur	Manager
Précondition	Authentification
Scénario nominal	-Accéder à l'espace « Projets ». -Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. -Vérifier l'existence d'un autre projet avec le même nom. -Vérifier la disponibilité du scrum master. -Enregistrer le projet dans la base de données.
Scénario d'échec (1)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». -Invalidité des champs remplis.
Scénario d'échec (2)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. -Existence d'un autre projet avec le même nom.
Scénario d'échec (3)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. -Vérifier l'existence d'un autre projet avec le même nom. -Indisponibilité du scrum master.
Postcondition	Affichage de la page du projet ajouté.

Tableau 19: Description des scénarios "Ajouter un projet"

La figure (56) illustre le diagramme de séquences du cas d'utilisation « Ajouter projet »

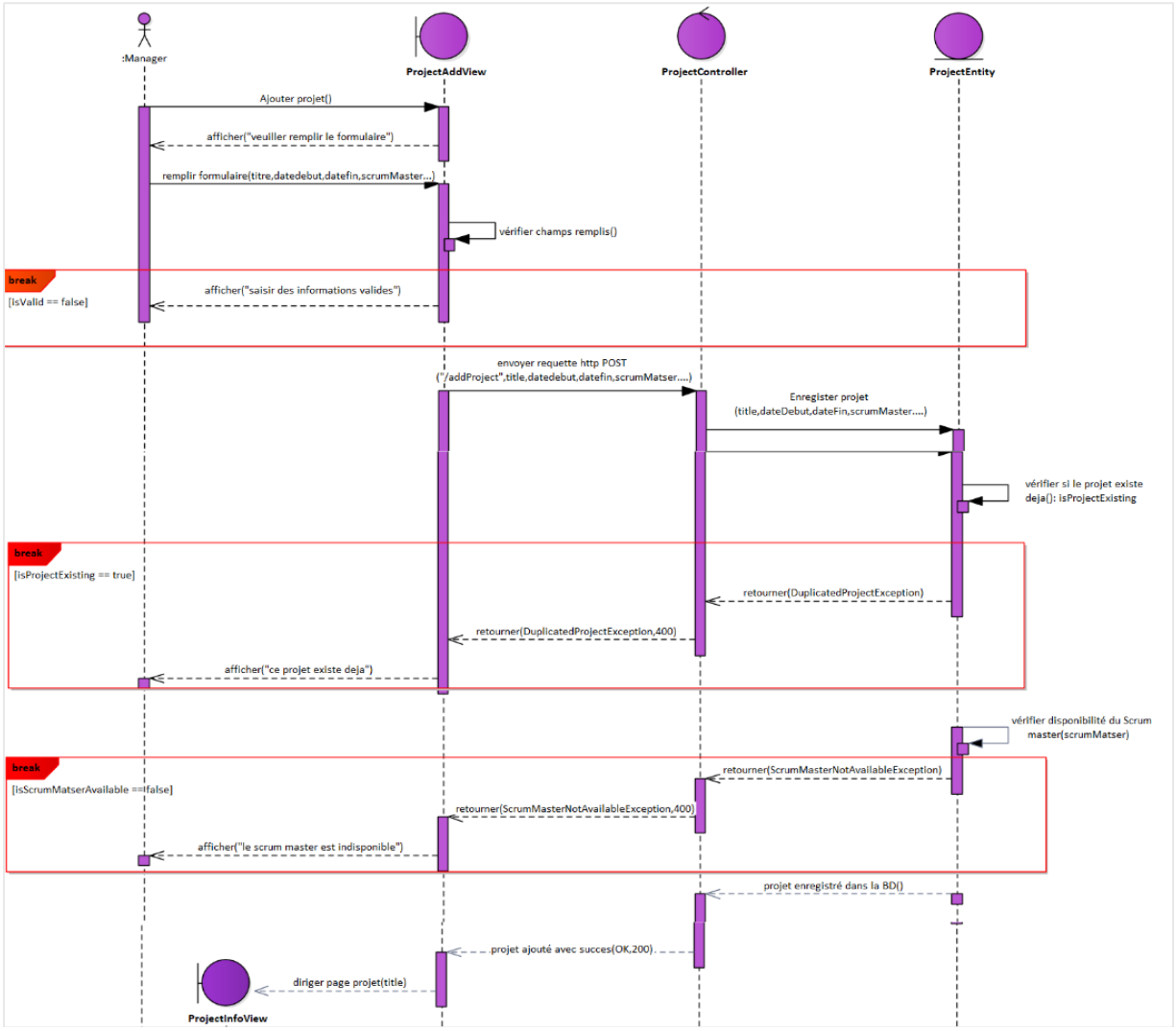


Figure 56: Diagramme de séquences <<Ajouter projet>>

II.5 Diagramme de séquences du cas d'utilisation : Créer product-backlog

Nom du cas d'utilisation	Créer product backlog
Acteur	Scrum master
Précondition	Authentification, Projet affectée
Scénario nominal	-Accéder à l'espace « Projets ». -Remplir le formulaire de création du backlog (titre,équipes). -Cliquer sur le bouton « Créer ». -Vérifier la validité des champs remplis. -Ajouter les fonctionnalités. -Enregistrer product backlog dans la base de données.
Scénario d'échec	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». **-Invalidité des champs remplis.**
Postcondition	Affichage d'un tableau du product backlog avec la liste des fonctionnalité ajoutées dans la même page du projet en question.

Tableau 20: Description des scénarios "Créer product-backlog"

La figure (57) illustre le diagramme de séquences du cas d'utilisation « Créer product-backlog ».

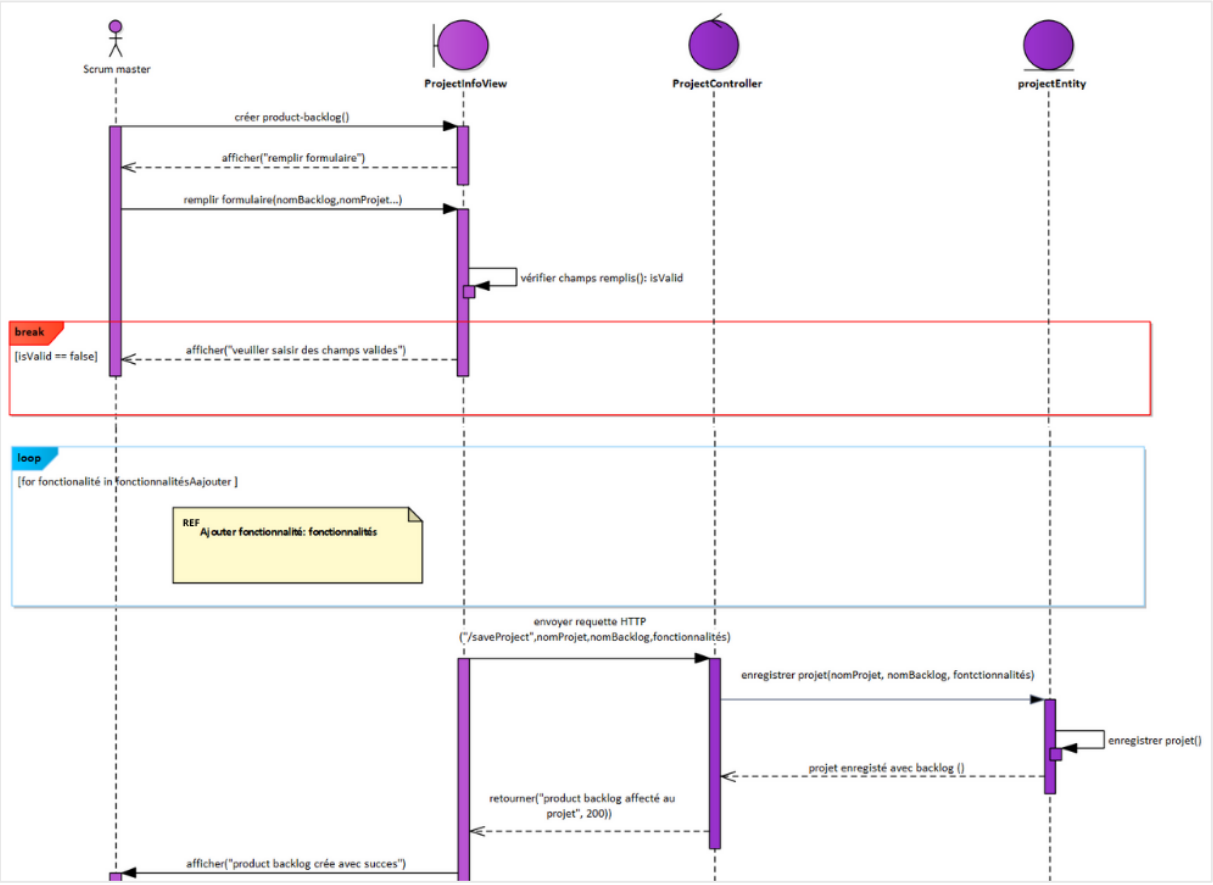


Figure 57: Diagramme de séquences <<Créer product-backlog>>

II.6 Diagramme de séquences du cas d'utilisation : Ajouter fonctionnalité

Nom du cas d'utilisation	Ajouter fonctionnalité
Acteur	Scrum master
Précondition	Authentification, projet affecté.
Scénario nominal	-Accéder à l'espace « Projets ». -Remplir le formulaire d'ajout d'une fonctionnalité (titre, description, complexité, priorité…). -Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. -Vérifier l'existence d'une autre fonctionnalité avec le même titre dans le projet en question. -Enregistrer la fonctionnalité dans la base de données.
Scénario d'échec (1)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». **-Invalidité des champs remplis.**
Scénario d'échec (2)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ».

	-Vérifier la validité des champs remplis. -Existence d'une autre fonctionnalité avec le même titre.
Postcondition	Fonctionnalités enregistrées.

Tableau 21: Description des scénarios "Ajouter fonctionnalité"

La figure (58) illustre le diagramme de séquences du cas d'utilisation « ajouter fonctionnalité ».

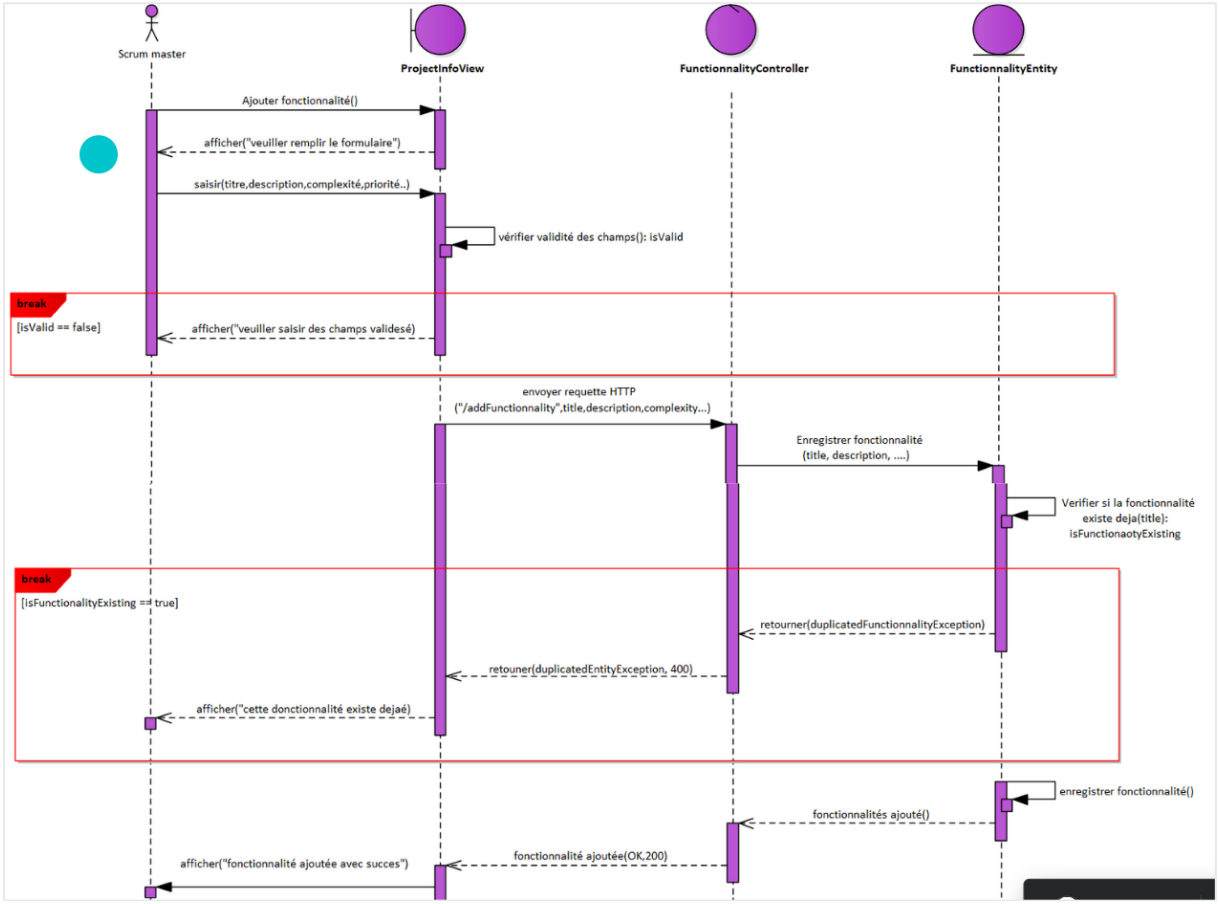


Figure 58: Diagramme de séquences <<Ajouter fonctionnalité>>

II.7 Diagramme de séquences du cas d'utilisation : planifier sprint

Nom du cas d'utilisation	Planifier sprint
Acteur	Chef d'équipe
Précondition	Authentification, projet affecté à son équipe.
Scénario nominal	- Accéder à l'espace « Projets ». -Remplir le formulaire d'ajout d'un sprint (titre, description, date début, date fin). -Cliquer sur le bouton « Planifier ». -Vérifier la validité des champs remplis. -Vérifier l'existence d'un autre sprint en cours dans le projet en question. -Initialiser l'état du sprint selon la date de début.

	-Enregistrer le sprint dans la BD. -Enregistrer le sprint dans le calendrier.
Scénario d'échec (1)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». -Invalidité des champs remplis.
Scénario d'échec (2)	-Remplir le formulaire d'ajout d'un projet (nom, titre, dateDebut, dateFinEstimée). -Cliquer sur le bouton « Ajouter ». -Vérifier la validité des champs remplis. -Existence d'un autre sprint en cours dans le projet en question.
Postcondition	Sprint planifié.

Tableau 22: Description des scénarios "Planifier sprint"

La figure (59) illustre le diagramme de séquences du cas d'utilisation « Planifier sprint ».

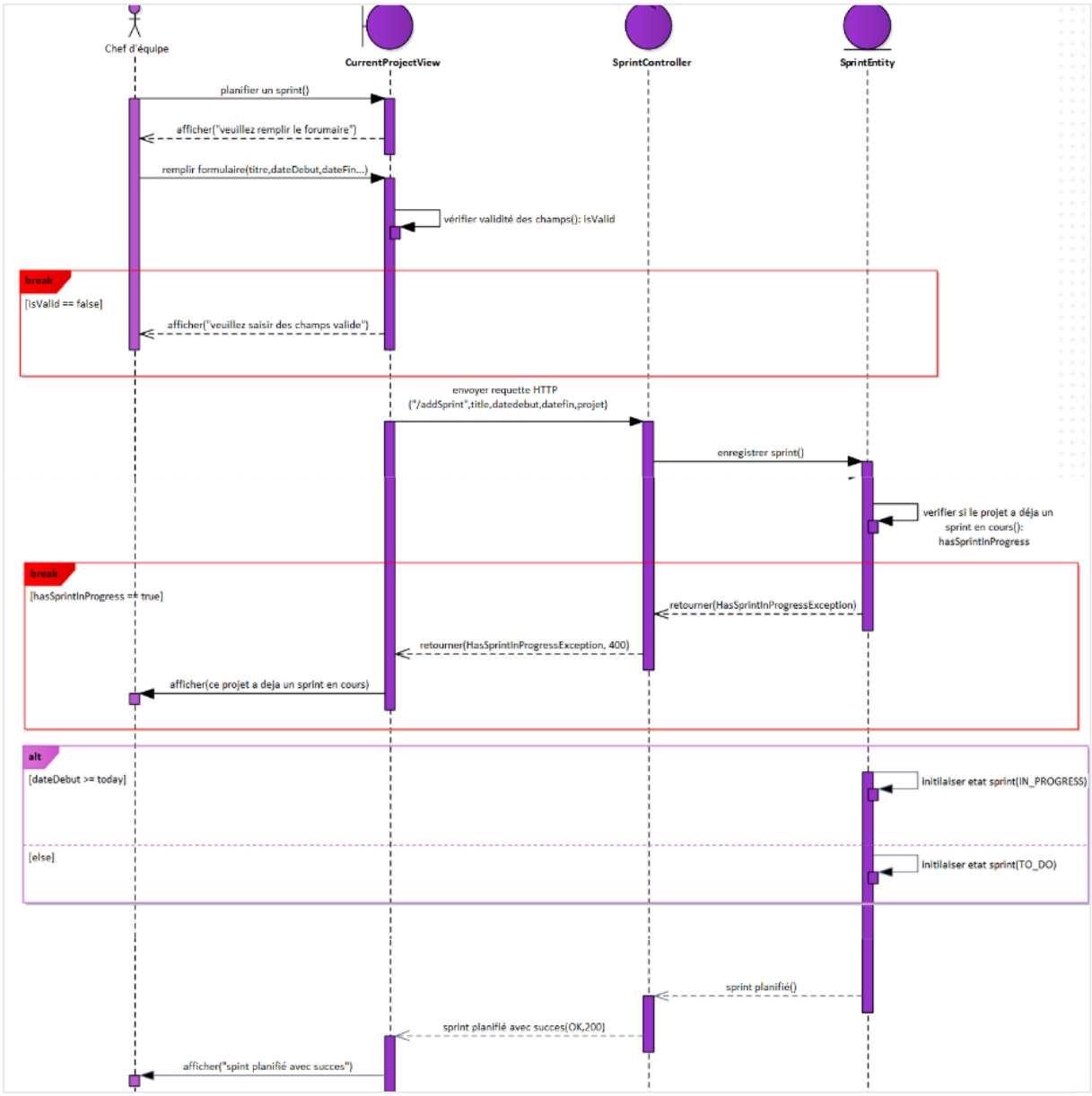


Figure 59: Diagramme de séquences <<Planifier sprint>>

II.8 Diagramme de séquences du cas d'utilisation : Terminer tâche

Nom du cas d'utilisation	Terminer tâche.
Acteur	Collaborateur.
Précondition	Authentification, tâche affectée.
Scénario nominal	-Accéder à l'espace « Tâches ». -Afficher drag and drop des taches. -Glisser la tache en question vers état terminé.
Postcondition	Tache indiquée comme terminée.

Tableau 23: Description des scénarios "Terminer tâche"

La figure (60) illustre le diagramme de séquences du cas d'utilisation « terminer tâche ».

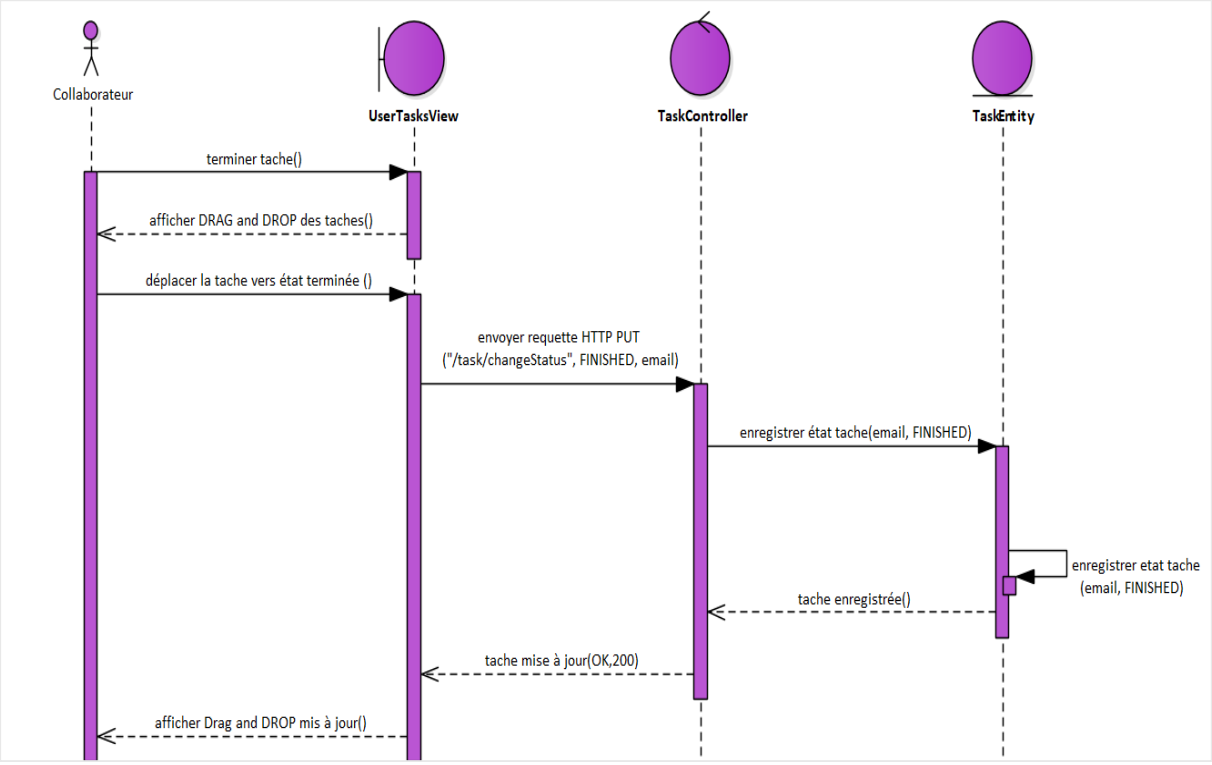


Figure 60: Diagramme de séquences <<Terminer tâche>>

II.9 Diagramme de séquences du cas d'utilisation : Envoyer une demande de congé

Nom du cas d'utilisation	Envoyer une demande de congé.
Acteur	Collaborateur.
Précondition	Authentification.
Scénario nominal	-Accéder à l'espace « Congés ». -Remplir le formulaire de demande de congé (date début, date fin, type, remplaçant,manager). -Cliquer sur le bouton « Envoyer ».

	-Vérifier la validité des champs remplis. -Vérifier l'insuffisance du solde de congé. -Vérifier la disponibilité du remplaçant. -Enregistrer la demande de congé dans la base de données.
Scénario d'échec (1)	-Accéder à l'espace « Congés ». -Remplir le formulaire de demande de congé. -Cliquer sur le bouton « Envoyer ». -Invalidité des champs remplis.
Scénario d'échec (2)	- Accéder à l'espace « Congés ». -Remplir le formulaire de demande de congé (date début, date fin, type, remplaçant). -Cliquer sur le bouton « Envoyer ». -Vérifier la validité des champs remplis.4-Insuffisance du solde de congé du demandeur.
Scénario d'échec (3)	-Accéder à l'espace « Congés ». -Remplir le formulaire de demande de congé (date début, date fin, type, remplaçant). -Cliquer sur le bouton « Envoyer ». -Vérifier la validité des champs remplis. -Vérifier l'insuffisance du solde de congé. -Indisponibilité du remplaçant.
Postcondition	Demande de congé envoyée au manager.

Tableau 24: Description des scénarios "Envoyer une demande de congé"

La figure (61) illustre le diagramme de séquences du cas d'utilisation « Envoyer demande de congé ».

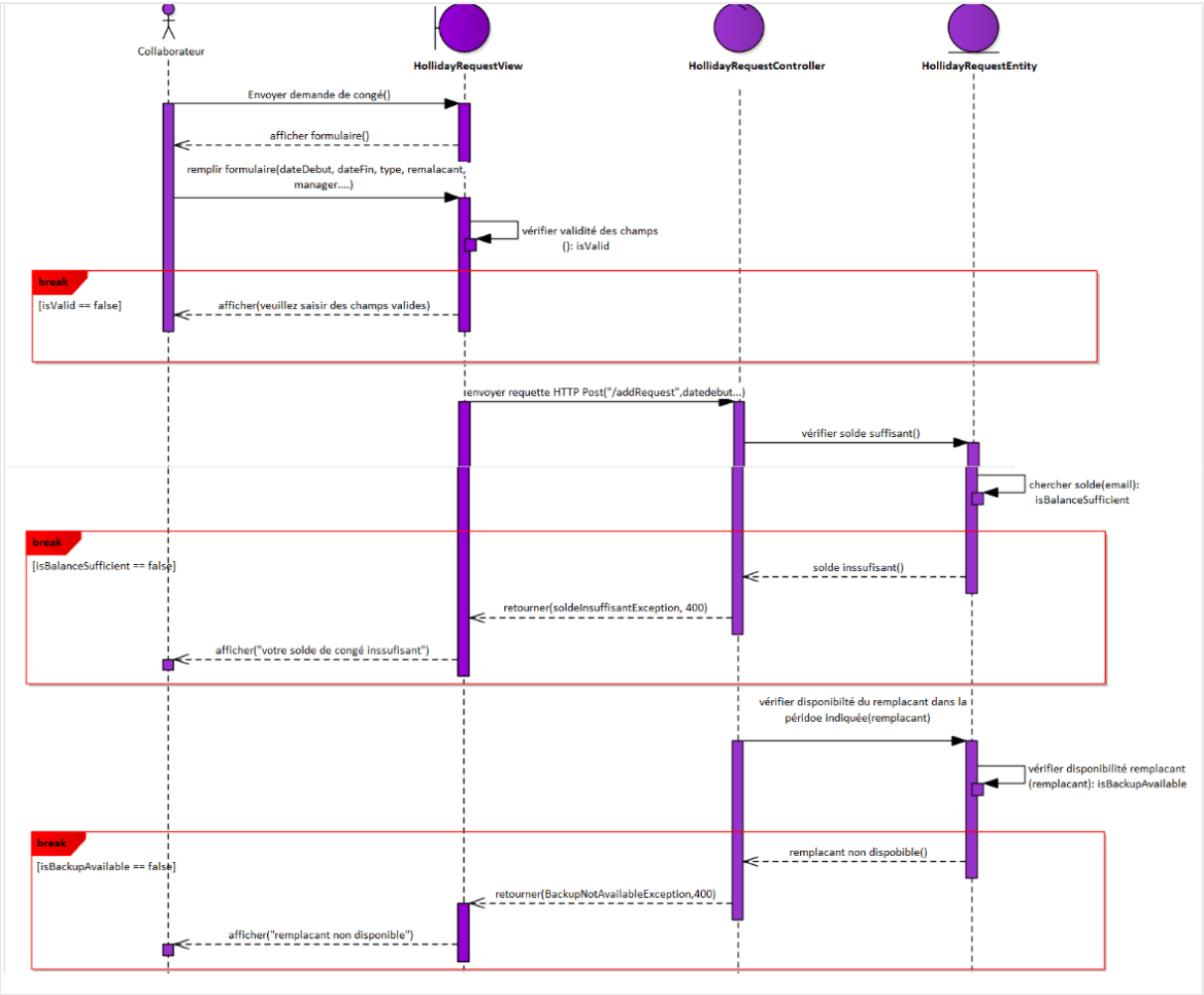


Figure 61: Diagramme de séquences <<Envoyer demande de congé>>

II.10 Diagramme de séquences du cas d'utilisation : Traiter une demande de congé

Nom du cas d'utilisation	Traiter une demande de congé.
Acteur	Manager.
Précondition	Authentification, demande de congé envoyé.
Scénario nominal	-Accéder à l'espace « Congé ». -Sélectionner la demande à approuver depuis un tableau. -Cliquer sur le bouton Approuver ou Rejeter. -Mettre à jour l'état de la demande selon la décision.
Postcondition	Demande de congé traitée.

Tableau 25: Description des scénarios "Traiter une demande de congé"

La figure (62) illustre le diagramme de séquences du cas d'utilisation « Traiter une demande de congé ».

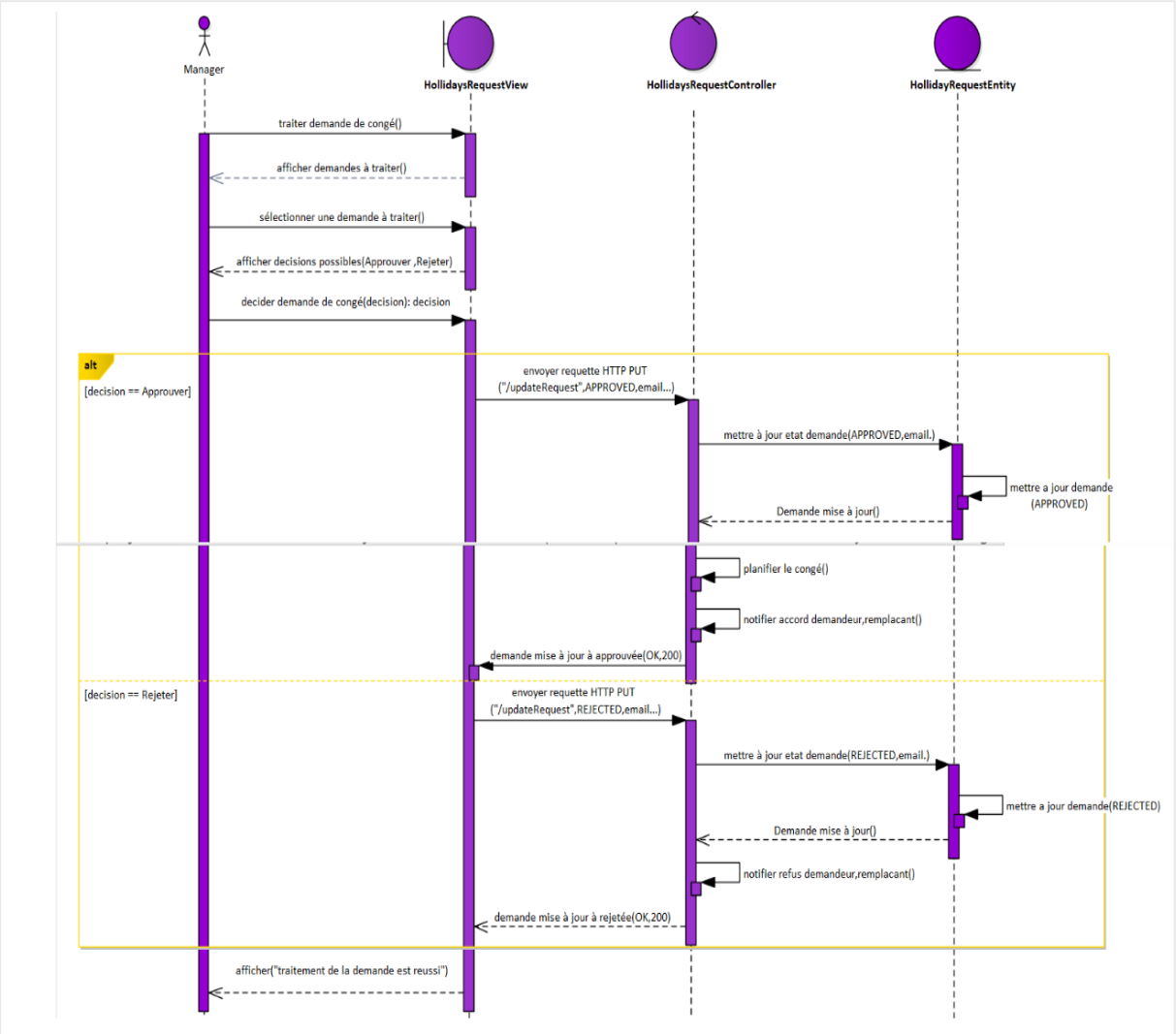


Figure 62: Diagramme de séquences <<Traiter demande de congé>>

III. Diagramme d'état-transition

Pour les diagrammes d'état-transition, nous avons sélectionné les classes entités les plus représentatives de notre projet. Ces diagrammes modéliseront ainsi de manière précise le comportement dynamique de ces classes.

III.1 Diagramme de d'état-transition de la classe : Collaborateur

Etat source	Transition	Etat cible
Au Bureau	Affecter une mission.	En Mission.
Au Bureau	Demande de congé approuvée.	En Congé.
En Congé	Durée de congé finie.	Au Bureau.
En Mission	Mission accomplie.	Au Bureau.

Au Bureau, En Congé, En Mission	Manager supprime le collaborateur.	Supprimée.

Tableau 26: Transitions d'état de la classe « Collaborateur »

La figure (63) illustre le diagramme d'état-transition de la classe « Collaborateur ».

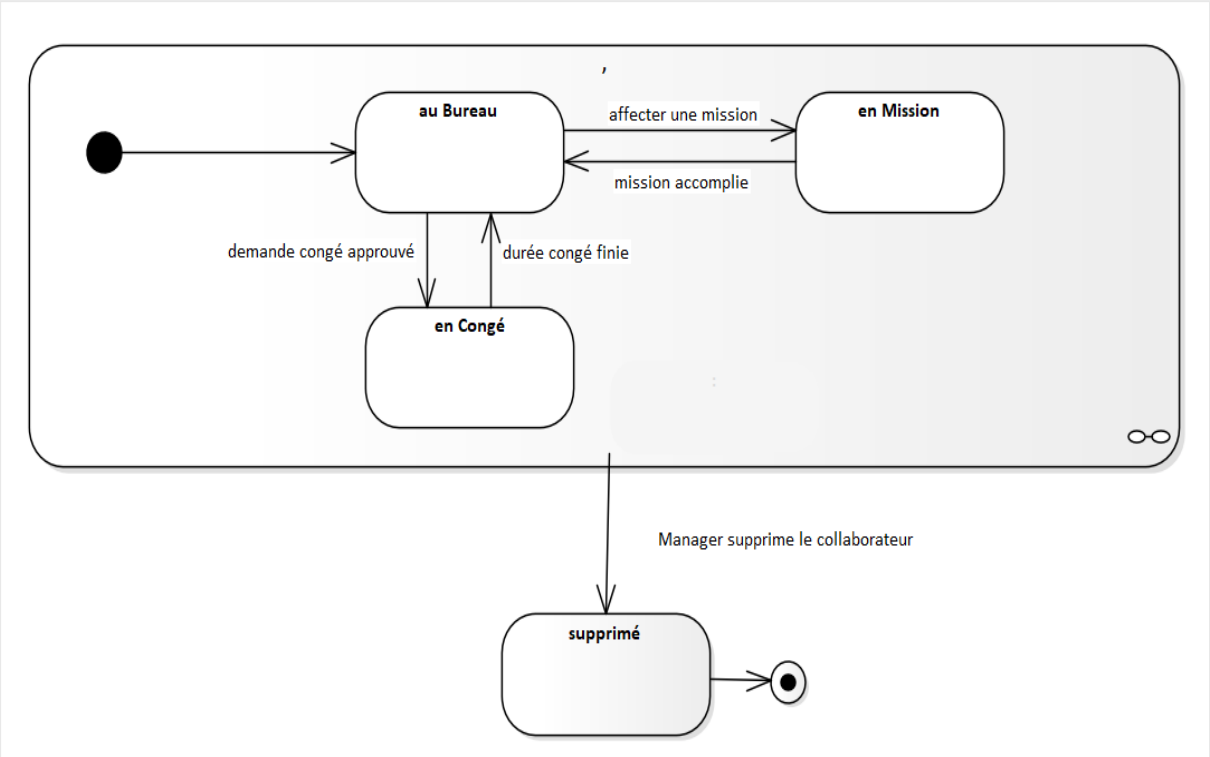


Figure 63: Diagramme d'état-transition de la classe <<Collaborateur>

III.2 Diagramme de d'état-transition de la classe : Projet

Etat source	Transition	Etat cible
A Faire	Date début arrivé à condition que le Manager annonce le début du projet.	En Cours.
A Faire	Date début arrivé et le manager n'a pas encore annoncé le début du projet.	Reporté.
Reporté	Manager annonce début du projet.	En cours.
En Cours	Date fin arrivé et le manager indique la fin du projet.	Terminé.
En cours	Date fin arrivé et le manager n'a pas encore indiqué fin projet.	En retard.

En Retard	Le manager indique fin du projet.	Terminé.
A Faire, En cours, En Retard, Terminé, Reporté	Le manager annule le projet.	Annulé.
Annulé	Le manager supprimer le projet.	Supprimé

Tableau 27: Transitions d'état de la classe « Projet »

La figure (64) illustre le diagramme d'état-transition de la classe « Projet ».

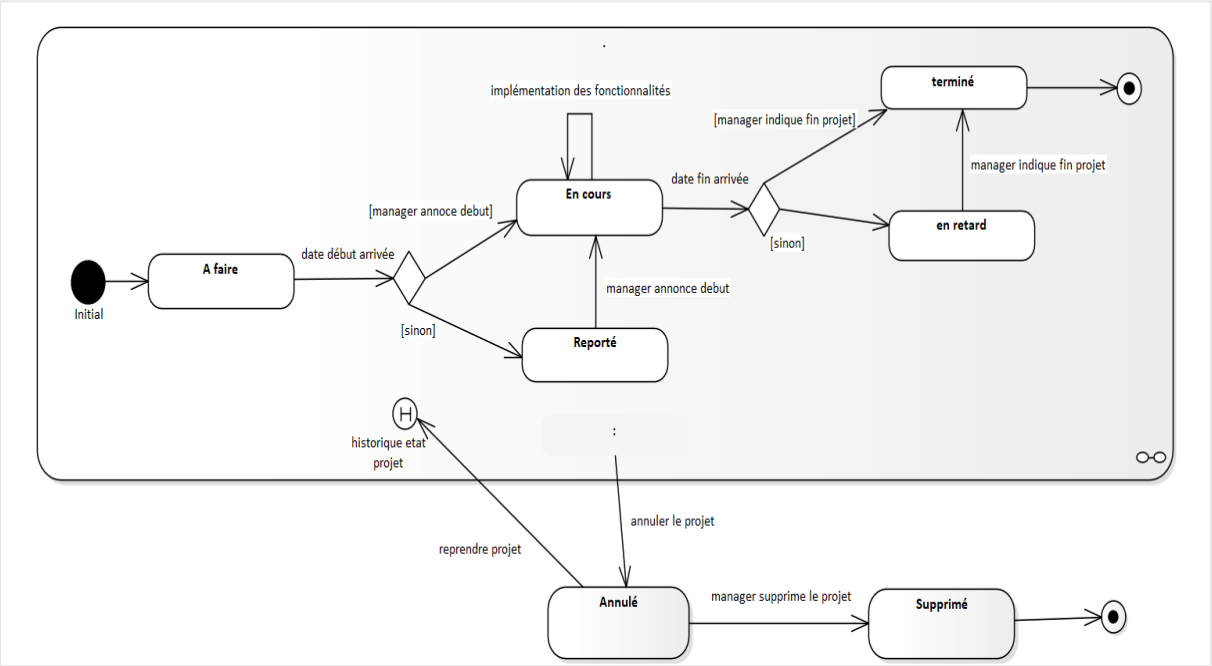


Figure 64: Diagramme d'état-transition de la classe <<Projet>

III.3 Diagramme de d'état-transition de la classe : Demande de congé

Etat source	Transition	Etat cible
En cours	Demandeur annule sa demande.	Annulée.
En cours	Manager approuve la demande.	Approuvée.
En cours	Manager rejette la demande.	Rejetée.

Tableau 28: Transitions d'état de la classe « Demande de congé »

La figure (65) illustre le diagramme d'état-transition de la classe « Demande de congé ».

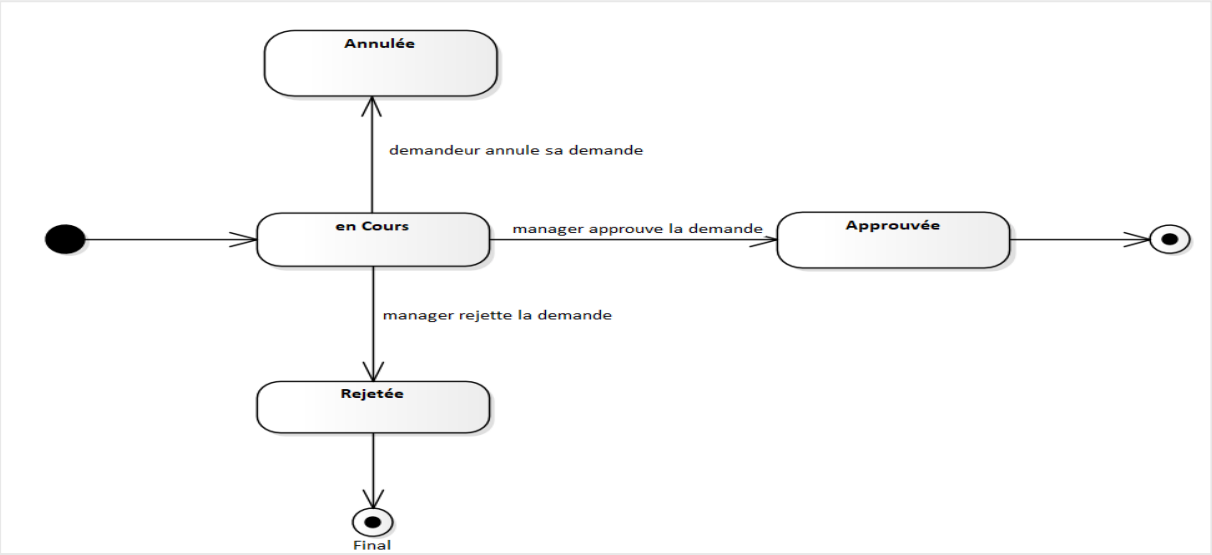


Figure 65: Diagramme d'état-transition de la classe <<Demande de congé>>

III.4 Diagramme de d'état-transition de la classe : Message

Etat source	Transition	Etat cible
Non Lu	Voir le message.	Lu.
Lu	Marquer message comme non lu.	Non Lu.
Lu, Non Lu	Supprimer le message.	Supprimé.

Tableau 29: Transitions d'état de la classe « Message »

La figure (66) illustre le diagramme d'état-transition de la classe « Message ».

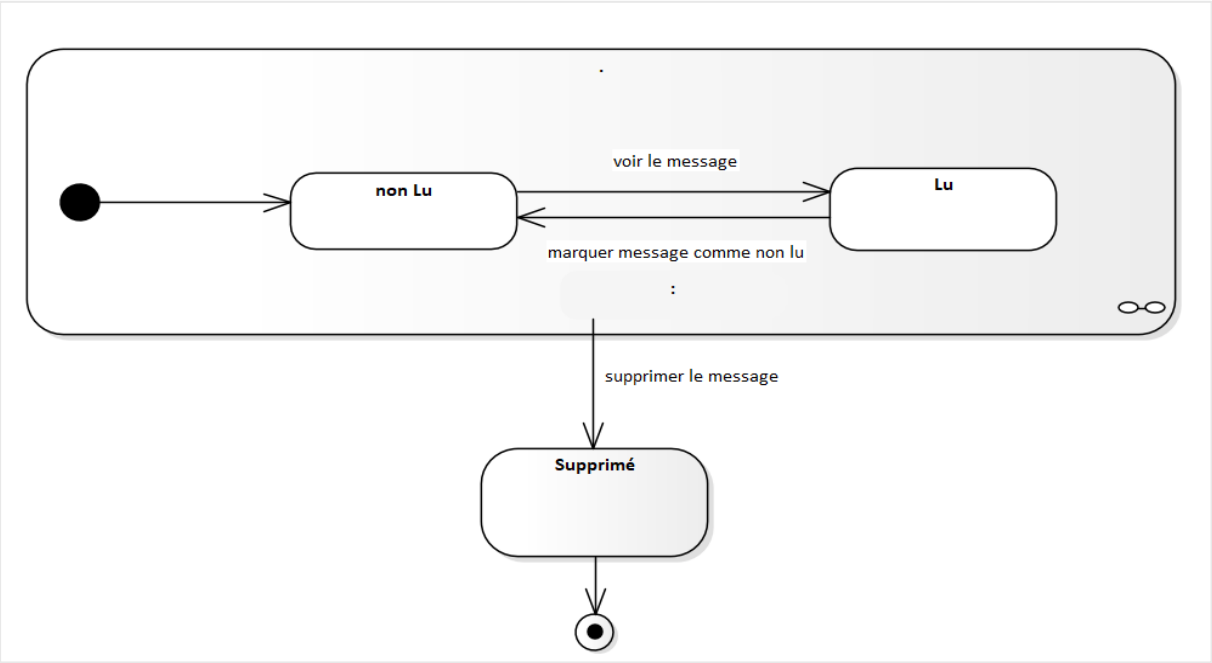


Figure 66: Diagramme d'état-transition de la classe <<Message>>

IV. Synthèse

Dans ce chapitre, nous avons détaillé la vue dynamique de notre système en élaborant les diagrammes de séquence pour les principaux cas d'utilisation et les diagrammes d'état-transition pour les classes d'entités à variation élevée. Cette combinaison a éclairé la vision dynamique et comportementale de notre application, assurant une compréhension approfondie de ses interactions et de son comportement global.

Le chapitre suivant se concentrera sur l'environnement technique de notre projet. Nous y détaillerons les différentes technologies utilisées, ainsi que les outils, langages et Frameworks qui ont été choisis pour le développement de notre application.

Chapitre VI : Environnement technique et architecture logicielle

Après avoir établi les différentes vues de la conception de notre système, ce chapitre se focalisera sur son environnement technique. Nous présenterons les outils de développement, les langages de programmation, les frameworks, les plugins et les systèmes de gestion de base de données que nous avons choisis pour notre projet. Nous justifierons nos choix en mettant en avant les avantages spécifiques de chaque technologie et leur pertinence par rapport à nos besoins.

En outre, nous détaillerons l'architecture logicielle du projet, y compris le pattern de décomposition des microservices, les communications inter-services, ainsi que l'architecture du chatbot.

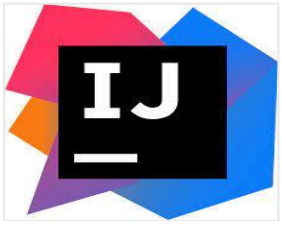
I. Environnement technique

Dans cette sous-partie, nous allons citer les différents outils de développement que nous avons utilisés, ainsi que la manière et les raisons de leur choix.

I.1 Outils de développement

Dans cette section, nous présenterons les outils de développement utilisés dans notre projet, notamment les IDE (Environnements de Développement Intégré), les logiciels et autres ressources essentielles pour la réalisation de notre application.

I.1.A Intellij idea



[13]

IntelliJ IDEA est un environnement de développement intégré (IDE) renommé et largement utilisé dans le domaine de la programmation. Développé par JetBrains, IntelliJ IDEA offre une multitude de fonctionnalités avancées et des outils puissants pour faciliter le processus de développement logiciel. **[13]**

Nous avons opté pour IntelliJ IDEA comme environnement de développement intégré (IDE) en raison de sa robustesse et de ses fonctionnalités avancées. Cet IDE offre une prise en charge exceptionnelle du langage Java, utilisé pour le backend de notre application. Les outils intégrés tels que le débogueur performant, la gestion des dépendances intuitive et les capacités de refactorisation automatique ont considérablement simplifié notre processus de développement.

I.1.B Visual studio code

[14]

Visual Studio Code (VS Code) est un environnement de développement intégré (IDE) extrêmement populaire et polyvalent, largement utilisé par les développeurs du monde entier. Conçu par Microsoft, VS Code offre une interface conviviale et une multitude de fonctionnalités avancées qui simplifient le processus de développement logiciel. **[14]**

Nous avons choisi Visual Studio Code comme éditeur de code pour le développement frontend de notre application. Visual Studio Code se distingue par sa légèreté et sa flexibilité, tout en offrant une riche panoplie de fonctionnalités et d'extensions. Son support étendu pour JavaScript et d'autres technologies web nous a permis de développer efficacement l'interface utilisateur.

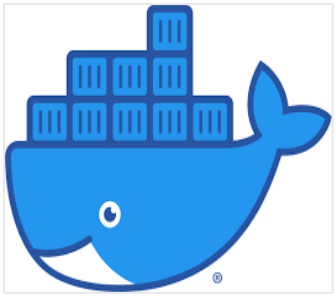
I.1.C Postman

[15]

Postman est un outil puissant et incontournable dans le domaine du développement d'API (Interfaces de Programmation d'Application). Il offre une plateforme complète pour créer, tester, documenter et déboguer des API, qu'elles soient internes ou externes. **[15]**

Nous avons choisi d'intégrer Postman dans notre processus de développement en raison de sa polyvalence. Cet outil nous permet de concevoir rapidement des requêtes HTTP pour interagir avec nos API, d'effectuer des tests automatisés, d'inspecter les réponses et de visualiser les données dans différents formats.

I.1.D Docker

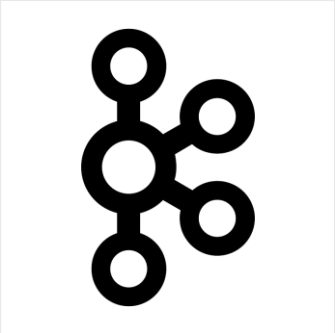


[16]

Docker propose une solution de conteneurisation qui permet d'emballer une application et tous ses dépendances dans un environnement isolé appelé conteneur. Ces conteneurs sont légers, portables et auto-suffisants, ce qui signifie qu'ils peuvent être exécutés de manière cohérente sur différentes plates-formes, que ce soit sur un ordinateur local, un serveur en cloud ou même un Datacenter. **[16]**

Cette décision découle de la nature de notre application, qui est développée selon une architecture orientée **microservices**. Parmi les diverses approches de déploiement des microservices, nous avons opté pour les conteneuriser, ainsi que leurs environnements, en un unique conteneur. Cette approche présente des avantages considérables tels que la **simplification du déploiement**, l'amélioration de la portabilité et la facilitation des tests.

I.1.E Apache kafka



[17]

Apache Kafka, une plateforme de streaming en temps réel, occupe une place de choix dans le domaine de l'architecture de microservices pour la communication événementielle. Son rôle clé réside dans la facilitation de la transmission et de la consommation de flux de données entre différentes parties d'une application, ce qui contribue à la création d'une architecture souple et résiliente. **[17]**

Dans notre projet, nous avons choisi Apache Kafka comme broker en raison de notre architecture microservices basée sur des événements. Les communications entre nos services, comme entre les services d'identité et de **notification,** ainsi que ceux de gestion des congés et de notification (etc.), reposent sur des messages asynchrones.

I.2 Langages de programmation

Ici, nous allons expliciter les langages de programmation utilisés lors de la phase de développement du projet, tant pour le backend que pour le frontend.

I.2.A Java

[18]

Java est un langage de programmation polyvalent et populaire qui a joué un rôle essentiel dans le domaine du développement logiciel depuis plusieurs décennies. Connu pour sa portabilité, sa fiabilité et sa vaste communauté de développeurs, Java offre un environnement propice à la création d'applications robustes et performantes. **[18]**

La motivation de choisir le langage Java réside dans la norme de développement adoptée par notre organisation. Près de **85% de nos projets** de développement web sont réalisés en **Java,** ce qui garantit une cohérence et une expertise approfondie au sein de notre équipe. Nous avons donc utilisé Java pour développer les différents microservices de notre projet, capitalisant ainsi sur cette expertise et assurant la cohérence avec les normes de développement de l'entreprise.

I.2.B Javascript

[19]

JavaScript est un langage de programmation essentiel dans le développement web moderne, apportant interactivité et dynamisme aux pages et aux applications. Son rôle fondamental en tant que langage de script côté client en fait un outil incontournable pour la création d'expériences utilisateur engageantes et interactives. **[19]**

Nous avons exploité JavaScript côté frontend de notre application pour plusieurs raisons. Tout d'abord, JavaScript est largement utilisé pour le développement frontend en raison de sa compatibilité avec les navigateurs web modernes. En utilisant JavaScript, nous pouvons créer des interfaces utilisateur interactives et dynamiques, offrant ainsi une expérience utilisateur fluide et réactive.

I.2.C Typescript

[20]

Typescript est un langage de programmation open-source développé par Microsoft. Il s'agit d'une surcouche de JavaScript qui ajoute des fonctionnalités telles que le typage statique, les classes, les interfaces, et bien plus encore. TypeScript permet aux développeurs de créer des applications JavaScript robustes, évolutives et maintenables. **[20]**

Nous avons choisi d'utiliser TypeScript dans notre projet en raison de ses nombreux avantages. Tout d'abord, TypeScript offre un **typage statique** qui permet de détecter les erreurs de typage dès la phase de développement, ce qui contribue à améliorer la robustesse et la fiabilité de notre code. De plus, TypeScript facilite la maintenance du code en fournissant une syntaxe plus expressive et en permettant une meilleure organisation du code grâce à l'utilisation de classes, d'interfaces et de modules. Enfin, en utilisant TypeScript avec Angular, le framework JavaScript que nous avons sélectionné pour notre projet, nous bénéficions d'une synergie puissante qui nous permet de développer des applications web modernes et performantes.

I.2.D Html & Css

[21]

HTML (HyperText Markup Language) et CSS (Cascading Style Sheets) sont deux langages fondamentaux pour le développement web. HTML est utilisé pour structurer le contenu d'une page web, définissant les différents éléments tels que les titres, les paragraphes, les listes, etc. CSS, quant à lui, est utilisé pour styliser et mettre en forme ces éléments HTML, en contrôlant les aspects visuels tels que les couleurs, les polices, les marges et les espacements**. [21]**

Dans le cadre de notre projet, nous avons exploité la combinaison HTML et CSS pour développer des interfaces utilisateur conviviales et esthétiquement attrayantes. En utilisant HTML, nous avons pu **structurer nos pages web** de manière logique, en définissant clairement les différents éléments et leur hiérarchie.

Ensuite, grâce à CSS, nous avons pu appliquer des styles visuels à ces éléments HTML, en créant une mise **en page harmonieuse et attrayante**. Nous avons défini des règles CSS pour contrôler les couleurs, les polices, les tailles et les espacements, assurant ainsi une présentation cohérente sur l'ensemble du site.

En combinant HTML et CSS de manière efficace, nous avons pu créer des interfaces utilisateur intuitives et conviviales, offrant une expérience utilisateur optimale à nos utilisateurs tout en respectant les normes de conception et de convivialité du web moderne.

I.3 Frameworks de développement

L'utilisation des frameworks dans notre projet était un choix stratégique pour faciliter la communication au sein de l'équipe de développement. Nous allons citer les principaux frameworks utilisés.

I.3.A Angular

[23]

Angular est un framework open-source développé par Google, largement utilisé pour la création d'applications web et mobiles à page unique (SPA). Il est basé sur TypeScript et offre une structure robuste pour le développement front-end, avec une architecture orientée composants, une gestion efficace de l'état de l'application et une large gamme de fonctionnalités prêtes à l'emploi. **[23]**

Nous avons choisi d'incorporer Angular dans notre stack technologique en raison de plusieurs facteurs. Tout d'abord, **la compatibilité des concepts d'Angular avec Java** est un avantage majeur pour notre équipe de développement, car notre backend est également développé en

Java. Cette compatibilité facilite l'intégration et la communication entre les différentes parties de notre application.

I.3.B Spring boot

[24]

Spring Boot est un Framework Java open-source conçu pour simplifier le développement d'applications Spring. Il offre des fonctionnalités puissantes pour créer rapidement des applications Java robustes et évolutives en utilisant une approche convention-over-configuration. Spring Boot facilite la configuration, la gestion des dépendances, et fournit un ensemble d'outils pour développer des applications web, des services RESTful, des microservices, et bien plus encore. **[24]**

Nous avons choisi d'utiliser Spring Boot dans notre projet en raison de sa popularité, de sa robustesse et de sa compatibilité avec notre **architecture microservices**. Malgré l'existence d'autres frameworks Java, Spring Boot s'est imposé comme le choix idéal pour notre projet en raison de sa facilité de développement et de déploiement en microservices. Grâce à ses fonctionnalités intégrées telles que l'inversion de contrôle, l'autoconfiguration et la gestion des dépendances, Spring Boot nous permet de créer des microservices de manière efficace tout en assurant une architecture modulaire et évolutive.

I.3.C Spring data jpa

[25]

Spring Data JPA est un projet de la suite Spring Data qui fournit une abstraction au-dessus de JPA (Java Persistence API). Il simplifie le développement d'applications Java en offrant une manière plus facile et plus fluide d'interagir avec les bases de données relationnelles à l'aide de JPA. Spring Data JPA simplifie les opérations CRUD (Create, Read, Update, Delete) en fournissant des méthodes de repository prêtes à l'emploi et en gérant automatiquement la création de requêtes SQL. **[25]**

Nous avons opté pour Spring Data JPA dans notre projet en raison de sa facilité d'utilisation et de sa compatibilité avec Spring Boot. En intégrant Spring Data JPA à notre application, nous bénéficions d'une couche d'abstraction supplémentaire qui simplifie l'accès aux données et réduit la quantité de code boilerplate nécessaire pour interagir avec la base de données. En utilisant les interfaces de repository fournies par Spring Data JPA, nous pouvons définir facilement des méthodes de requête personnalisées et **effectuer des opérations CRUD sans avoir à écrire de requêtes SQL manuellement.** Cela nous permet d'améliorer la productivité du développement et de garantir la cohérence et la qualité de nos accès aux données dans toute l'application.

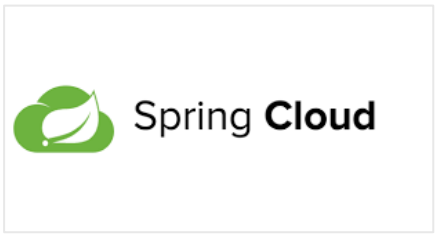
I.3.D Spring Security

[26]

Spring Security est un projet de la suite Spring qui fournit des fonctionnalités de sécurité pour les applications Java. Il permet de gérer l'authentification, l'autorisation, la protection contre les attaques CSRF (Cross-Site Request Forgery), les attaques XSS (Cross-Site Scripting), etc. Spring Security propose une architecture flexible et modulaire qui facilite l'intégration de fonctionnalités de sécurité avancées dans les applications Java. **[26]**

Dans notre application, nous avons exploité **JSON Web Tokens (JWT)** fourni par Spring Security pour mettre en œuvre l'authentification stateless. Cette approche permet à notre application de gérer l'état de l'authentification côté client de manière sécurisée et efficace. Grâce à Spring Security, nous pouvons garantir que notre application est protégée contre les menaces de sécurité tout en offrant une expérience utilisateur fluide et sécurisée.

I.3.E Spring Cloud



[27]

Spring Cloud est un ensemble de bibliothèques et de Frameworks développés par Spring pour faciliter la création d'applications basées sur une architecture de microservices. Il fournit des outils essentiels pour gérer la découverte des services, la configuration distribuée, la tolérance aux pannes, la gestion de la charge et bien plus encore, tout en simplifiant la complexité inhérente aux environnements de microservices. **[27]**

Nous avons décidé d'incorporer Spring Cloud dans le développement de notre application en raison de sa capacité à répondre aux exigences de notre architecture microservices. En adoptant l'architecture de microservices, il est essentiel d'avoir des mécanismes de gestion et de communication entre nos services. Spring Cloud offre une solution complète à cet égard, avec des composants tels **qu'Eureka pour la découverte de microservices,** Config Server pour la gestion de la configuration distribuée, et bien d'autres encore.

I.3.F Spring AI

[28]

Spring AI est un Framework d'application conçu pour l'ingénierie de l'IA. Son objectif est d'appliquer aux domaines de l'IA les principes de conception de l'écosystème Spring, tels que la portabilité et la conception modulaire, et de promouvoir l'utilisation de POJO (Plain Old Java Objects) comme éléments constitutifs d'une application dans le domaine de l'IA. **[28]**

Dans le cadre de notre application, nous avons exploité les capacités du modèle **GPT-4** fourni par OpenAI pour concevoir notre chatbot. Grâce à l'intégration de ce modèle, nous avons développé **un chatbot RAG** (Retrieval Augmented Generation). Cette intégration nous a permis de fournir des fonctionnalités avancées de traitement du langage naturel et de génération de texte. Ainsi, notre chatbot peut désormais engager des interactions plus sophistiquées et pertinentes sur le plan contextuel.

I.4 Plugins et dépendances

Nous avons également intégré divers plugins et dépendances pour optimiser notre projet, améliorer l'efficacité du développement, et enrichir les fonctionnalités de l'application. Les principaux plugins et dépendances utilisés seront détaillés ci-après.

I.4.A Lombok

[30]

Lombok est une bibliothèque Java qui permet de réduire la quantité de code boilerplate (code redondant et verbeux) nécessaire pour créer des classes Java. Elle offre des annotations simples à utiliser pour générer automatiquement des méthodes telles que les getters, les setters, les constructeurs et les méthodes equals/hashCode, ainsi que pour gérer la logique de la journalisation. **[30]**

Nous avons intégré Lombok dans notre projet pour simplifier le développement et améliorer **la lisibilité de notre code** Java. Grâce à ses annotations, nous avons pu réduire la quantité de code nécessaire pour définir nos classes, ce qui nous a permis de nous concentrer davantage sur la logique métier de notre application plutôt que sur des tâches répétitives et fastidieuses.

I.4.B Swagger

[32]

Swagger est un outil populaire utilisé pour la documentation des API. Il fournit une interface utilisateur interactive qui permet de visualiser et de tester facilement les différentes API d'une application. Grâce à ses fonctionnalités, Swagger permet de générer automatiquement une documentation claire et détaillée sur les endpoints, les paramètres et les réponses attendues. **[32]**

Dans notre projet, nous avons choisi d'exploiter Swagger pour **documenter nos API** afin de faciliter la communication au sein de l'équipe de développement. En fournissant une documentation à jour et interactive, Swagger permet à chaque membre de l'équipe de comprendre rapidement le fonctionnement des API et de collaborer de manière plus efficace.

De plus, l'interface utilisateur intuitive de Swagger facilite les tests et les validations des API, ce qui contribue à accélérer le processus de développement et à garantir la qualité des services exposés.

I.4.C Zipkin

[34]

Zipkin est un système de suivi distribué open-source qui permet de surveiller et de déboguer les requêtes distribuées à travers les différents composants d'une application. Il fournit des informations détaillées sur la façon dont les requêtes traversent les différents services et aide à identifier les goulots d'étranglement et les problèmes de performance. **[34]**

Dans notre projet, nous avons intégré Zipkin pour surveiller la **traçabilité des requêtes** à travers nos services microservices. Grâce à Zipkin, nous pouvons suivre le chemin emprunté par chaque requête depuis son point d'entrée jusqu'à sa sortie, en identifiant les services impliqués et les temps de réponse de chacun.

I.4.D Prometheus & Grafana

[35]

Prometheus est un système de surveillance open-source conçu pour surveiller les métriques et les alertes des applications et des systèmes. Grafana, de son côté, est une plateforme open-source de visualisation de données qui permet de créer des tableaux de bord et des graphiques pour visualiser les données collectées par Prometheus. **[35]**

Dans notre projet, nous avons utilisé Prometheus et Grafana pour assurer **l'observabilité** de nos microservices. Prometheus collecte les métriques et les indicateurs de performance de nos services, tandis que Grafana les visualise sous forme de tableaux de bord et de graphiques

intuitifs. Cette combinaison nous permet de surveiller en temps réel la santé et les performances de nos microservices, d'identifier les tendances et les anomalies, et de prendre des mesures proactives pour résoudre les problèmes avant qu'ils n'affectent les utilisateurs finaux.

I.4.E Websockets

[36]

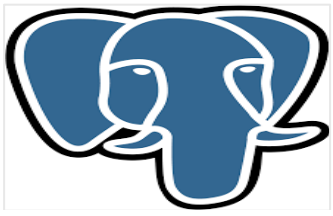
Les WebSockets sont un protocole de communication qui permet une communication bidirectionnelle et en temps réel entre un navigateur web et un serveur. Contrairement aux requêtes HTTP traditionnelles, qui suivent un modèle de requête-réponse, les WebSockets établissent une connexion persistante qui permet au serveur d'initier des communications avec le client à tout moment. **[36]**

Dans notre projet, nous avons choisi d'utiliser les WebSockets, en particulier pour le microservice de notification "notification-service". Ce service utilise les WebSockets pour **notifier en temps réel** les collaborateurs de l'organisation sur divers événements ou mises à jour importantes, tels que les demandes de congé, les affectations de projets, etc.

I.5. Systèmes de gestion de base de données (SGBD)

Nous allons maintenant citer et détailler les principaux SGBD employés dans notre projet en justifiant nos choix et en expliquant les besoins spécifiques auxquels chacun répondait.

I.5.A PostgreSQL



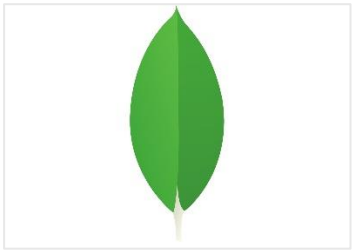
[37]

PostgreSQL est un système de gestion de base de données relationnelle (SGBDR) open source et puissant, largement utilisé pour stocker et gérer des données structurées. Il offre de nombreuses fonctionnalités avancées, telles que la prise en charge des transactions ACID

(Atomicité, Cohérence, Isolation, Durabilité), des jointures complexes, des vues, des déclencheurs, des fonctions stockées, et bien plus encore. **[37]**

Dans le cadre de notre projet, nous avons choisi PostgreSQL comme base de données principale en raison de sa fiabilité, de ses performances élevées et de sa capacité à gérer efficacement les données structurées. De plus, PostgreSQL prend en charge **la vectorisation des données,** ce qui est particulièrement pertinent pour notre application de chatbot RAG (Retrieval Augmented Generation).

I.5.B MongoDB



[38]

MongoDB est un système de gestion de base de données NoSQL orienté document, conçu pour stocker, interroger et gérer des données semi-structurées et non structurées. Il offre une grande flexibilité et évolutivité, ainsi qu'une prise en charge native de la réplication, de la distribution géographique et du partitionnement des données. **[38]**

Dans notre projet, MongoDB est sélectionné pour plusieurs microservices, notamment « forum-service » et « patrimoineDoc-service ». Le forum-service permet aux collaborateurs de partager des informations et de discuter de divers sujets, nécessitant une gestion flexible et rapide des données en temps réel. MongoDB répond à ces exigences grâce à sa capacité de stockage évolutive et à son indexation puissante, assurant une gestion efficace des discussions.

II. Architecture de l'application

Cette deuxième partie a pour objectif d'argumenter le choix de l'architecture et d'expliquer son fonctionnement.

II.1 Motivations du choix de l'architecture

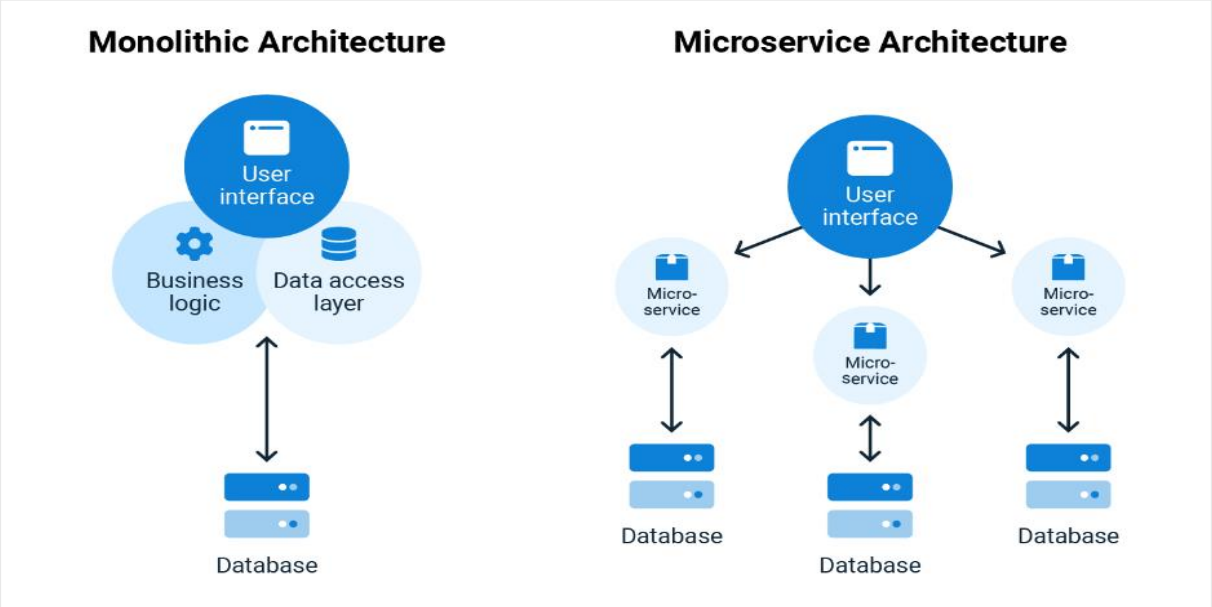


Figure 67: Choix de l'architecture logicielle ***[10]***

Nous avons opté pour une architecture **basée sur les microservices** pour notre projet, car elle répond parfaitement aux exigences fonctionnelles et non fonctionnelles précédemment mentionnées. Les raisons de ce choix sont les suivantes :

➢ **Scalabilité :** La nature modulaire des microservices permet de mettre à l'échelle indépendamment chaque composant de notre application en fonction des besoins spécifiques. Par exemple, si le nombre de collaborateurs de l'organisation utilisant notre application augmente soudainement, nous pouvons augmenter le nombre d'instances du microservice responsable de la gestion des utilisateurs sans affecter les autres parties de l'application.

➢ **Résilience :** En isolant les services les uns des autres, notre application devient plus résiliente. Par exemple, si le microservice gérant l'authentification tombe en panne, l'utilisateur ne sera pas bloqué, car une deuxième instance du service répondra à sa requête. De cette manière, les autres services peuvent continuer à fonctionner normalement.

➢ **Évolutivité :** Par exemple, un composant dédié à la découverte de services et à l'équilibrage de charge s'assurera que chaque service peut gérer les charges de travail variables de manière optimale, garantissant ainsi une flexibilité accrue pour répondre aux exigences de l'application.

➢ **Portabilité :** Les microservices permettent une grande portabilité, facilitant la réutilisation des composants dans différentes applications. Par exemple, nous pouvons réutiliser le microservice du chatbot dans une autre application déployée sur les serveurs de notre organisation.

II.2 Pattern de décomposition : Business Capability

Dans notre approche de décomposition des microservices, nous adoptons une stratégie basée sur **les besoins métier**, connue sous le nom de **Business Capability**. Cette approche consiste à décomposer notre application en différents microservices en fonction des différentes capacités métier de l'organisation.

Chaque microservice est conçu pour répondre à un besoin métier spécifique, ce qui permet une plus grande cohérence et une meilleure gestion des fonctionnalités de l'application. Par exemple, nous pouvons avoir des microservices dédiés à la gestion des projets, au suivi des congés, à la communication interne, ou à d'autres aspects métier essentiels. Cette décomposition par capacité métier nous permet de concevoir une architecture modulaire et flexible, capable de s'adapter aux évolutions des besoins de l'organisation tout en maintenant une haute cohérence et un faible couplage entre les différents composants de l'application.

Le tableau (30) présente les microservices identifiés et le besoin métier visé par ce dernier.

Microservice	*Objectif*	*Type*	*Responsable*
identity-service	Gestion de l'identité des collaborateurs et améliorer la connaissance de l'existant.	Métier	M.KHOUNA Younes
projectScrum-service	Gestion des projets dans le cadre du Framework Scrum.	Métier	M.KHOUNA Younes
hollidays-service	Traitement des demandes des congés.	Métier	M.KHOUNA Younes
feelingBoard-service	Analyse du sentiment quotidien des collaborateurs.	Métier	M.KHOUNA Younes
chatbotRAG-service	Aide et assistance.	Métier	M.KHOUNA Younes
Notification-service	Notification en temps réel.	Métier	M.KHOUNA Younes
chatRoom-service	Communication intra-équipe.	Métier	M.IBRAHIMI amine
privateChat-service	Communication intra-collaborateur.	Métier	M.IBRAHIMI amine
patrimoineDoc-service	Gestion des documents.	Métier	M.IBRAHIMI amine
Forum-service	Partages des activités des collaborateurs.	Métier	M.IBRAHIMI amine

Eureka-registration-service	Scalabilité, Résilience et Équilibrage de charge de l'application.	Technique	M.KHOUNA Younes + M.IBRAHIMI amine
Proxy-service	Routage des requêtes du client http.	Technique	M.KHOUNA Younes + M.IBRAHIMI amine
ConfigCloud-service	Externalisation de la configuration des microservices dans le cloud (github).	Technique	M.KHOUNA Younes + M.IBRAHIMI amine

Tableau 30: Décomposition des microservices

II.3 Architecture globale de l'application

Après avoir identifié les microservices métier et techniques de notre système, nous avons élaboré l'architecture globale de l'application ainsi que les interactions possibles entre les différents microservices identifiés.

En effet, nous avons utilisé exclusivement des **communications asynchrones** (également connues sous le nom Event Driven Architecture EDA) ce qui permet de garantir la résilience de notre application. Ces communications seront détaillées dans la partie suivante.

La figure (68) illustre l'architecture logicielle de notre application.

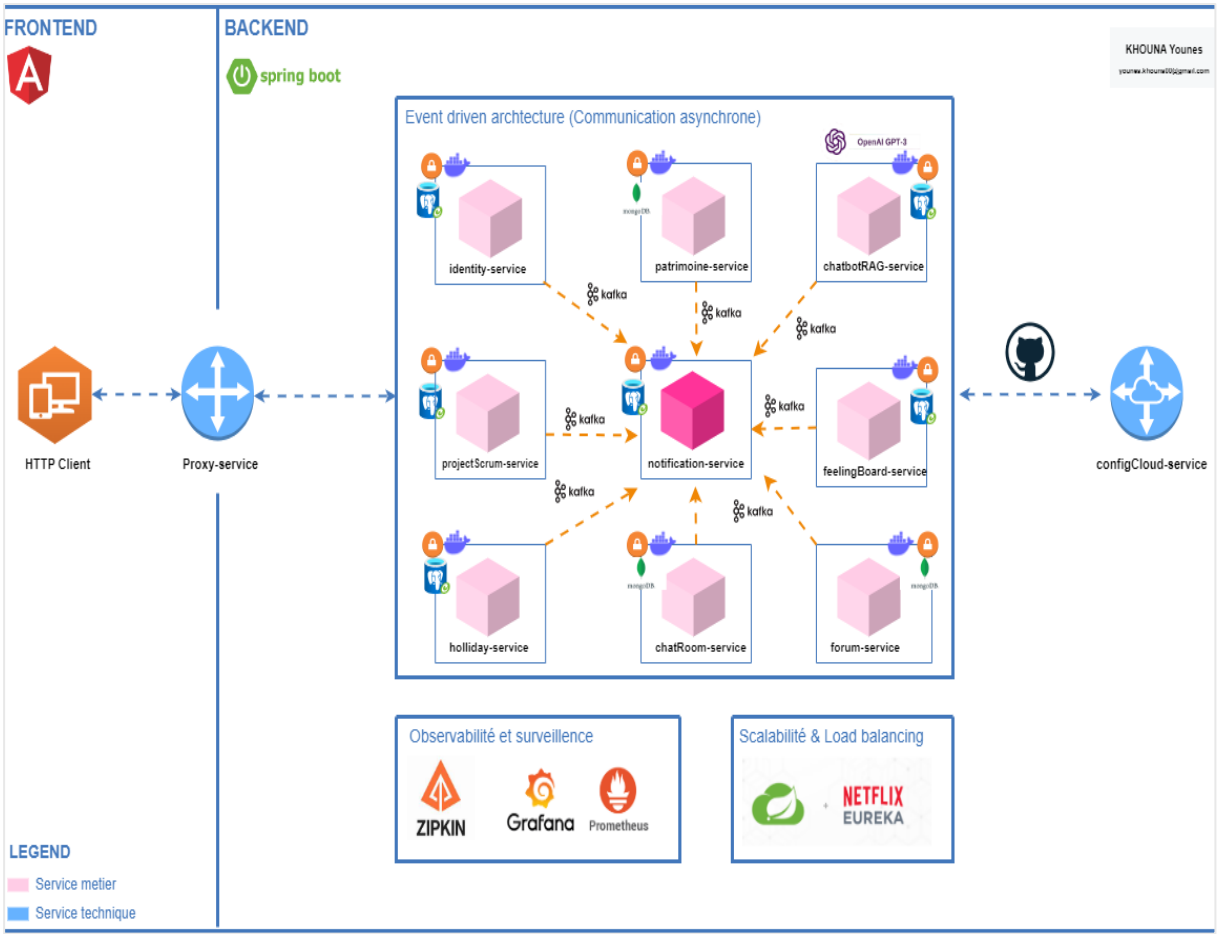


Figure 68: Architecture de l'application

II.4 Communication inter-services EDA (Event Driven Architecture)

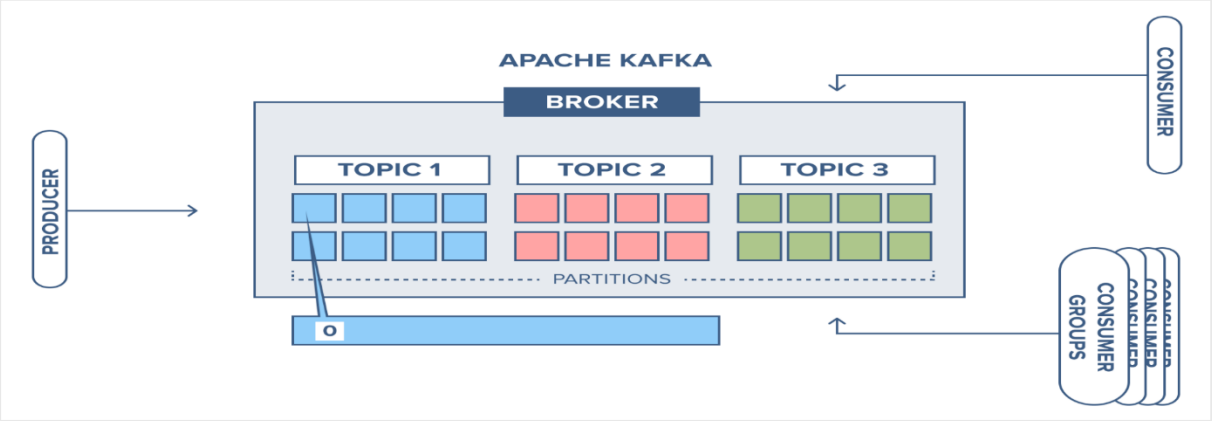


Figure 69: Architecture Appach kafka ***[11]***

Concernant les communications asynchrones que nous avons établies, elles se déroulent principalement entre le microservice gérant les notifications (« notification-service ») et tous les autres microservices métier de notre application. Nous avons exploité la robustesse d'**Apache Kafka** pour gérer les événements, assurant ainsi une transmission fiable et évolutive des messages.

Pour cette partie, nous allons illustrer un exemple réel de communications asynchrones inter-services. Nous décrirons en détail un scénario qui déclenche chacune de ces communications.

II.4.A Etude de cas : Communication entre « identity-service » et « notification-service »

Pour cette communication, le « identity-service » joue le rôle du producteur Kafka (Kafka producer), tandis que le « notification-service » est le consommateur Kafka (Kafka consumer) qui écoute tous les événements des sujets (topics) auxquels il est abonné.

Un scénario concret est décrit comme suit :

➢ **Kafka consumer :** le « notification-service » est abonné au sujet (topic) intitulé « newUserAdded » pour l'ajout d'un nouveau collaborateur.

➢ **Kafka producer :** Un manager ajoute un nouveau collaborateur dans la plateforme « DXC Synergy » ainsi le « identity-service » envoie cet évènement au Kafka-Broker ainsi que les données en question sous format JSON.

➢ **Kafka broker :** le kafka broker suite à l'évènement d'ajout d'un collaborateur qui a déclenché le sujet **« newUserAdded »** va ainsi notifier le consommateur « notification-service » par cet évènement.

➢ **Kafka consumer :** le « notification-service » reçoit les données en question et notifie les managers du département et les collègues de ce nouveau collaborateur par cet évènement.

La figure (70) illustre tous les scénarios possibles de la communication asynchrone entre « identity-service » et « notification-service »

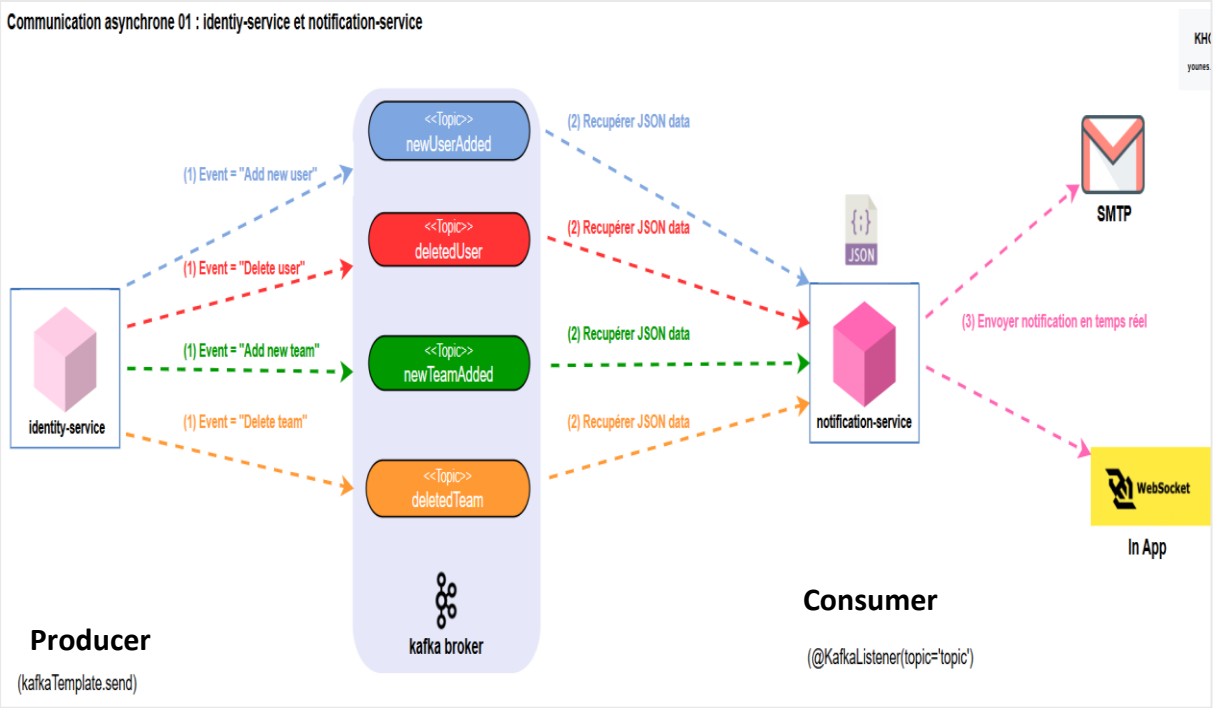


Figure 70: Communication asynchrone identity-service et notification-service

II.6 Architecture du chatbot RAG

Dans le cadre de notre projet, nous allons exploiter les « Large Language Models » **(LLM),** et plus précisément le modèle **GPT-4 d'OpenAI**, pour créer un chatbot performant capable de répondre aux questions des collaborateurs de l'organisation en s'appuyant sur les données des référentiels de l'entreprise.

Cette architecture se décompose en plusieurs phases distinctes à savoir :

➢ **Ingestion :** La première phase consiste à téléverser la base de connaissances de l'entreprise. Cette base de données comprend tous les documents et informations nécessaires que le chatbot utilisera pour répondre aux questions des utilisateurs.

➢ **Décomposition (Splitting) :** Une fois les documents téléversés, ils sont découpés en petits morceaux pour faciliter le traitement et l'analyse. Cette étape est cruciale pour

améliorer la précision des réponses fournies par le chatbot, car elle permet de traiter des portions plus gérables de texte.

➢ **Vectorisation :** Chaque morceau de document découpé est ensuite vectorisé. Cela signifie qu'il est transformé en vecteurs numériques à l'aide de techniques de traitement du langage naturel. Cette transformation permet au système de comprendre et de manipuler les données textuelles de manière plus efficace.

➢ **Stockage :** Les vecteurs résultants sont ensuite stockés dans une base de données spécialement conçue pour gérer ces représentations numériques. Cette base de données permet un accès rapide et efficace aux informations nécessaires pour générer des réponses.

➢ **Interaction utilisateur :** Une fois ces étapes complétées, le client est prêt à poser des questions via l'application DXC Synergy. Le chatbot utilise les vecteurs stockés pour rechercher les informations pertinentes et génère des réponses adaptées aux requêtes des collaborateurs.

La figure (71) montre l'architecture du chatbot RAG de notre application.

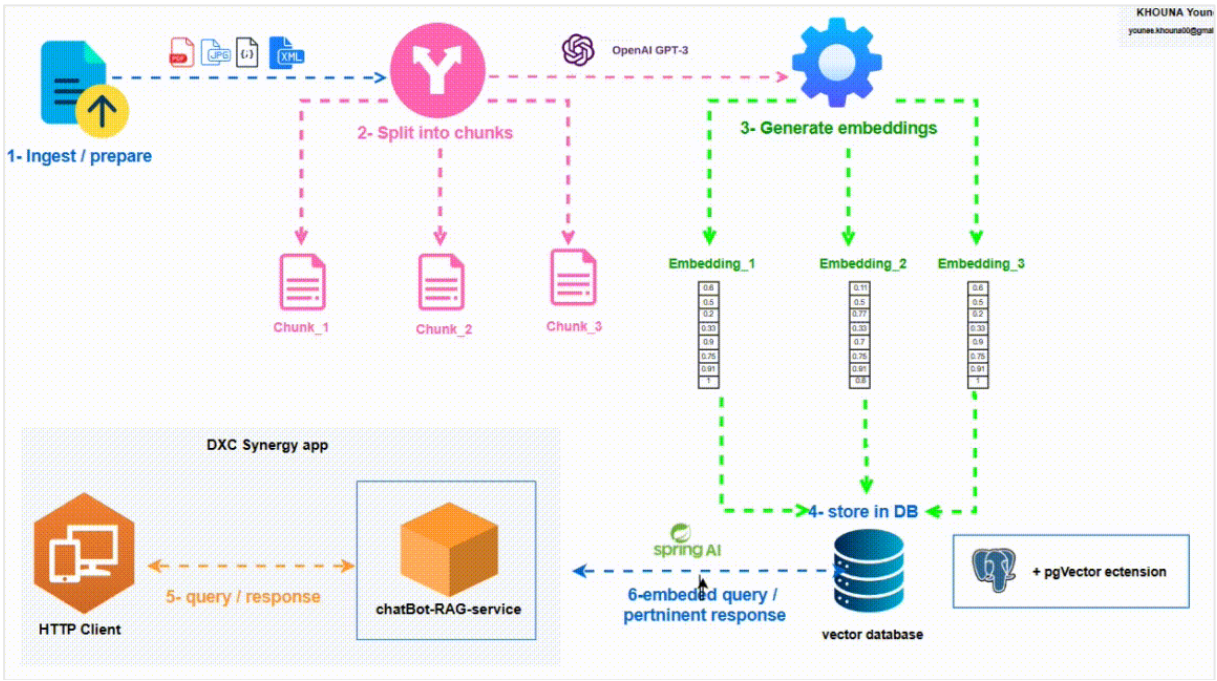


Figure 71: Architecture du chatbot RAG

III. Synthèse

Dans ce chapitre, nous avons présenté deux aspects essentiels de notre projet. Dans la première partie, nous avons exploré l'environnement technique, détaillant les outils de développement, les langages de programmation, les Frameworks, les plugins et les systèmes de gestion de base de données que nous avons sélectionnés. Nous avons justifié ces choix en mettant en avant les avantages spécifiques de chaque technologie et leur pertinence par rapport à nos besoins. Dans la seconde partie, nous avons examiné l'architecture logicielle de notre projet, en décrivant en détail le pattern de décomposition des microservices, les communications inter-services et le fonctionnement du chatbot.

Le prochain chapitre sera consacré à la réalisation de notre application. Nous y présenterons des captures d'écran illustrant les différentes fonctionnalités et interfaces que nous avons développées, offrant ainsi une vision concrète de notre travail et des résultats obtenus.

Chapitre VII : Réalisation

Après avoir détaillé l'environnement technique en mentionnant les différents outils, langages, et Frameworks de développement, nous sommes désormais prêts à aborder la phase de réalisation du projet. Ce chapitre présentera la réalisation du projet selon deux perspectives : celle de l'utilisateur final et celle du développeur.

I. Point de vue utilisateur

Dans cette partie, nous allons présenter une démonstration de l'application à travers plusieurs scénarios d'utilisation.

I.1 Scénario : Nouveau projet affecté

➢ **Authentification du manager :** ici le **manager, M. KHOUNA Younes**, du département des systèmes d'information DSI va s'authentifier pour accéder à son espace.

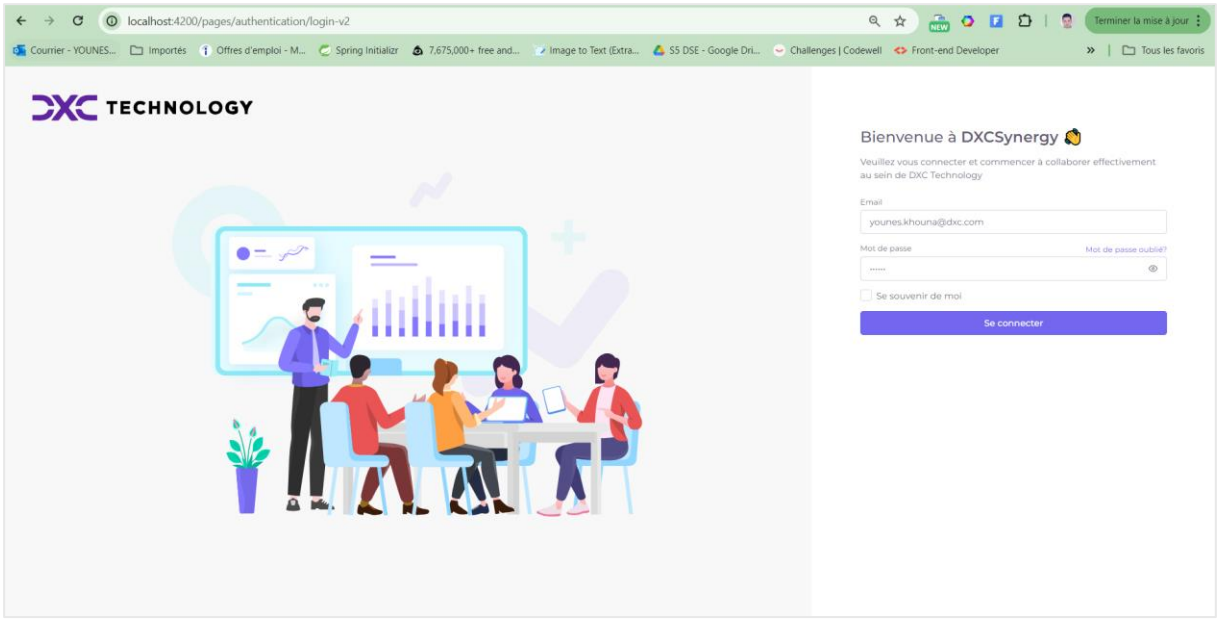


Figure 72: Page d'authentification

➢ **Accès à l'espace « Accueil » :** Une fois authentifié, notre manager est redirigé vers l'espace d'accueil, où il peut consulter toutes les informations importantes du jour **(le 31/05/2024).**

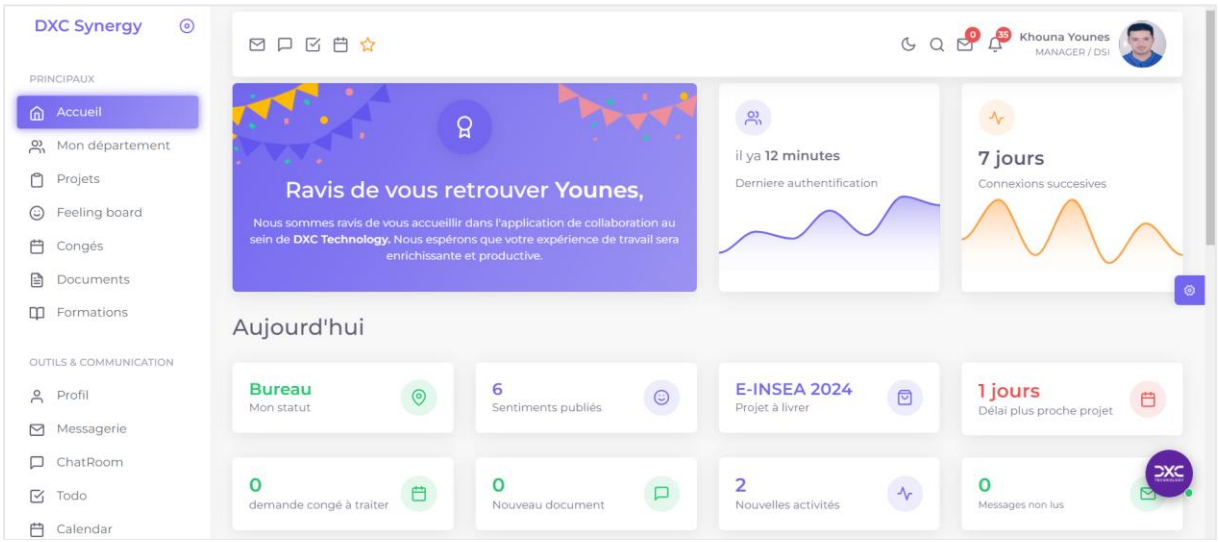


Figure 73: Page espace<<Accueil>> vue Manager 1/2

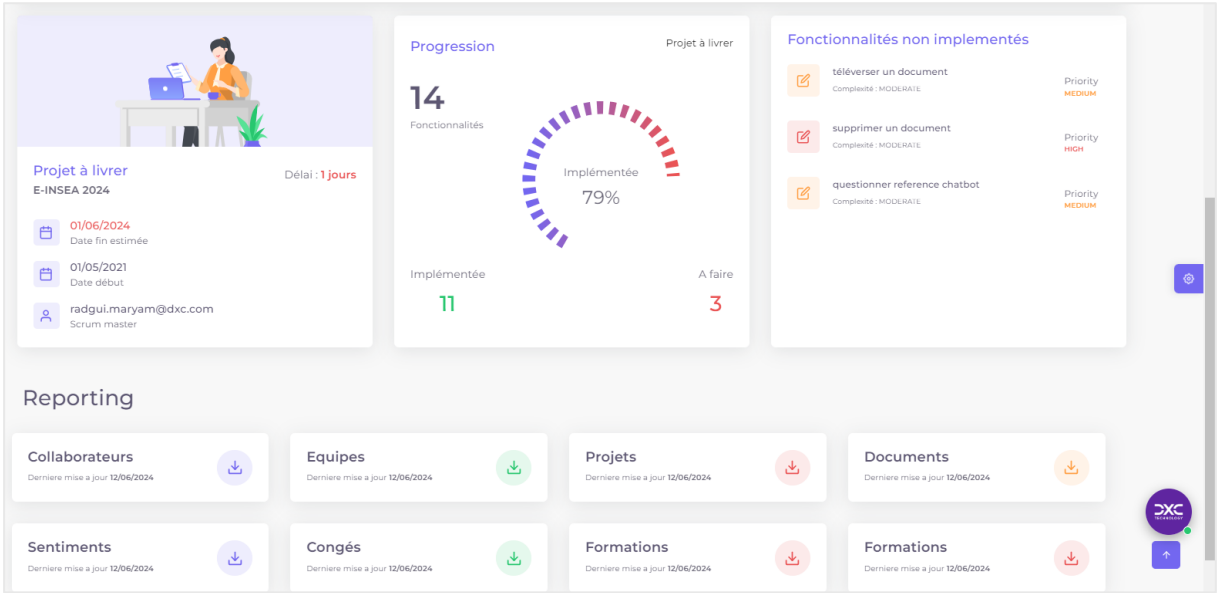


Figure 74: Page espace<<Accueil>> vue Manager 2/2

➢ **Accès à l'espace « Projets » :** Pour ajouter un nouveau, notre manager doit accéder à l'espace projet.il va ainsi voir tous les projets avec la possibilité de les filtrer.

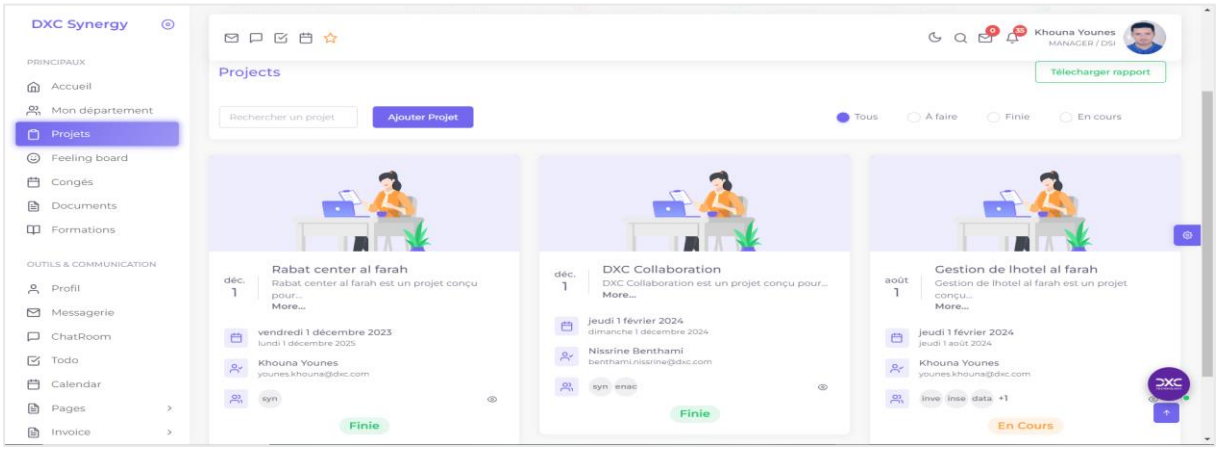


Figure 75: Page espace<<Projets>> vue Manager ½

➢ **Ajout d'un nouveau projet :** Pour ajouter un nouveau projet, il suffit de cliquer sur le bouton « Ajouter projet » et remplir le formulaire en question. Dans ce scénario notre manager a sélectionné **BELFAHMI Ahmed** comme étant le **Scrum master** du projet.

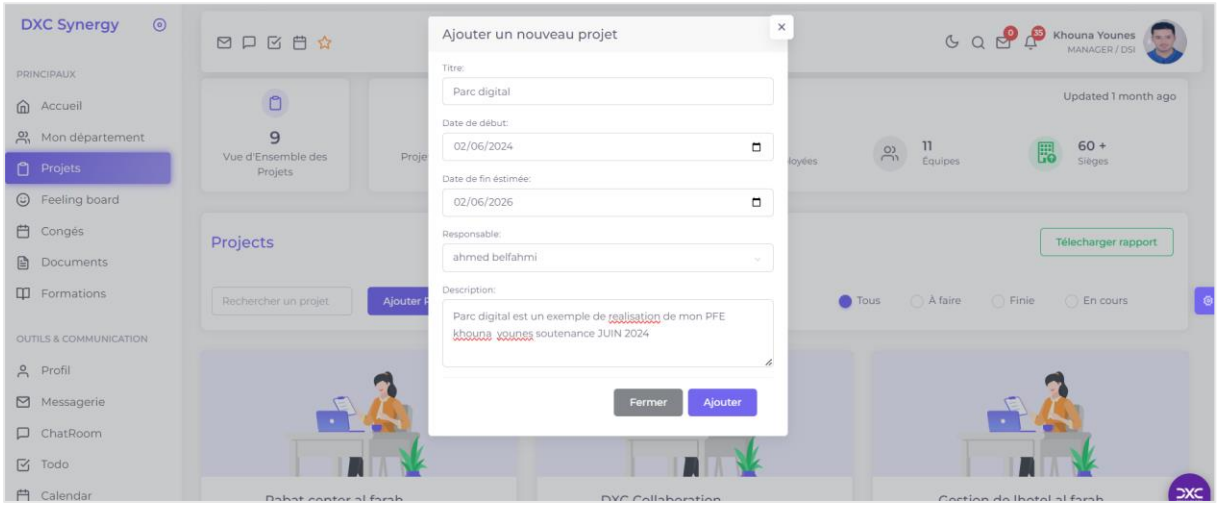


Figure 76: Formulaire d'ajout d'un projet

➢ **Réception de la notification pour Scrum master :** Après avoir été désigné comme Scrum Master du projet, **M. BELFAHMI Ahmed** recevra une notification de cette affectation.

NB : nous avons utilisé notre propre email pour les tests des notifications SMTP.

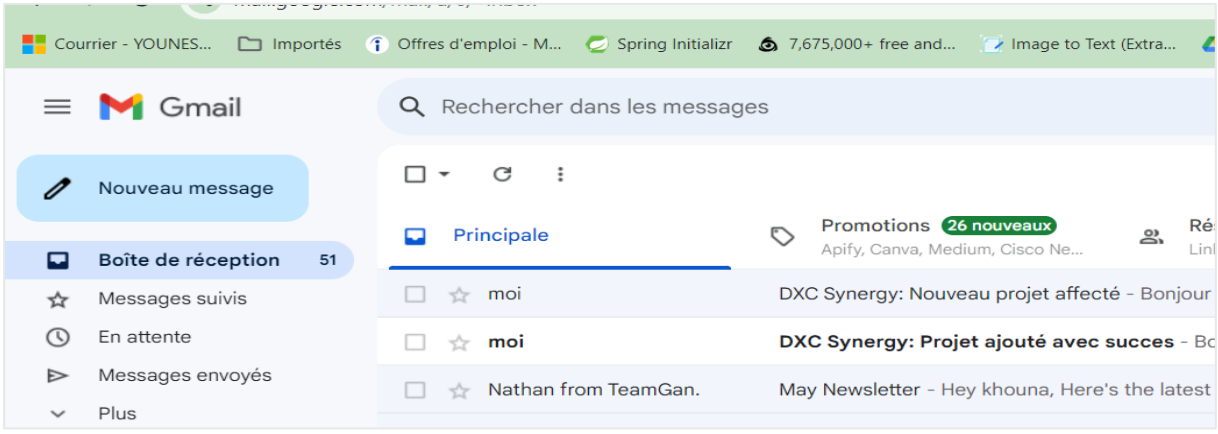


Figure 77: Réception de la notification via SMTP pour Scrum master

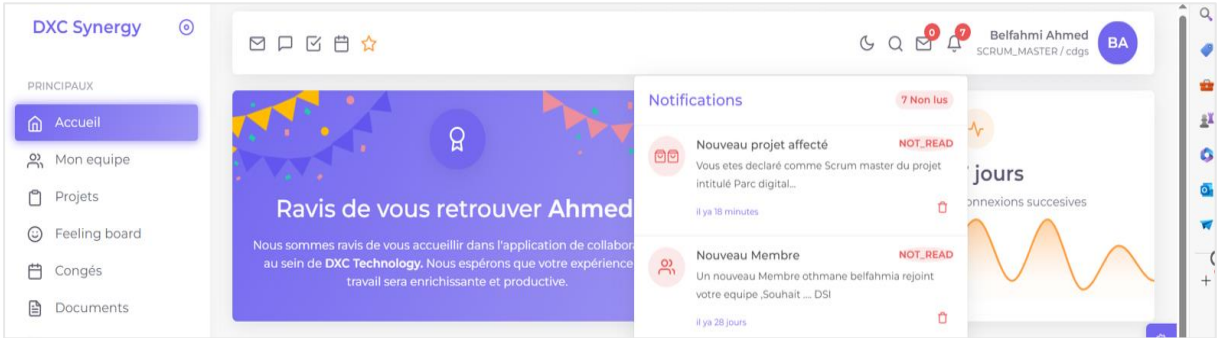


Figure 78: Réception de la notification dans l'application pour Scrum master

➢ **Création du product-backlog et sélection des équipes :** Maintenant, notre scrum master peut créer le product-backlog tout en sélectionnant les équipes concernées puis il ajoute les fonctionnalités requises.

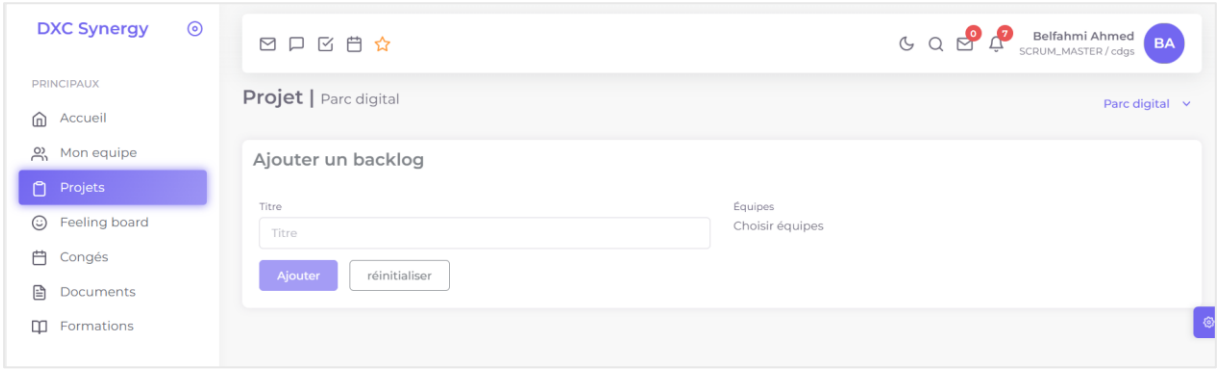


Figure 79: Création du product-backlog

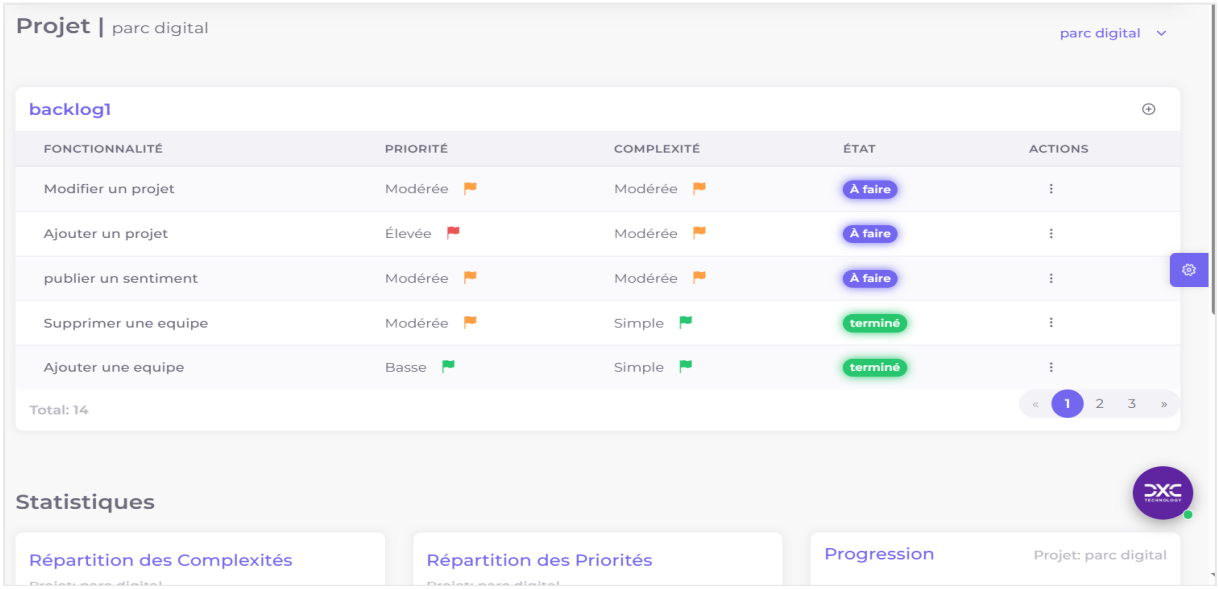


Figure 80: Ajout des fonctionnalités du projet

➢ **Notification des équipes :** Une fois une équipe est sélectionné dans un projet par le scrum master, tous les membres de celle-ci recevront une notification (**Ali est l'un des membres**).

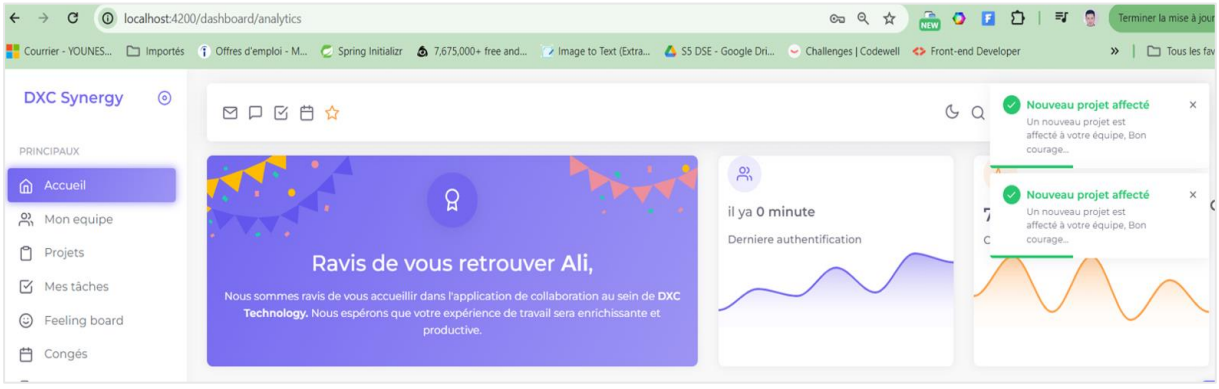


Figure 81: Notification des équipes "nouveau projet"

➢ **Planification des sprints par le chef d'équipe :** Dans ce scénario l'équipe DXC Synergy est affectée au projet en question, ainsi le **chef** de celle-ci à savoir **Mme AIT OUARAB Ouissal** peut planifier les sprints tout en affectant les taches pour chaque membre de l'équipe.

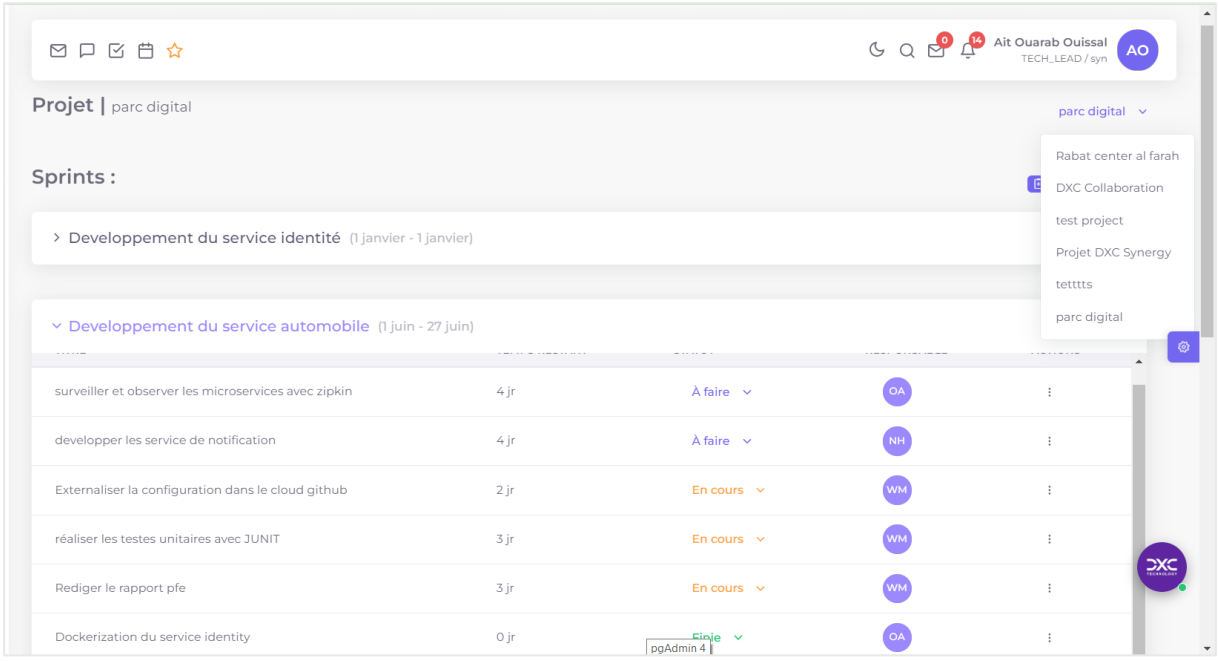


Figure 82: Planification des sprints par le chef d'équipe

➢ **Suivie des tâches par le chef d'équipe :** Après avoir réparti les tâches entre les membres de l'équipe, y compris lui-même, le chef d'équipe peut consulter l'état des tâches et, si nécessaire, modifier l'état d'une tâche assignée à un autre membre.

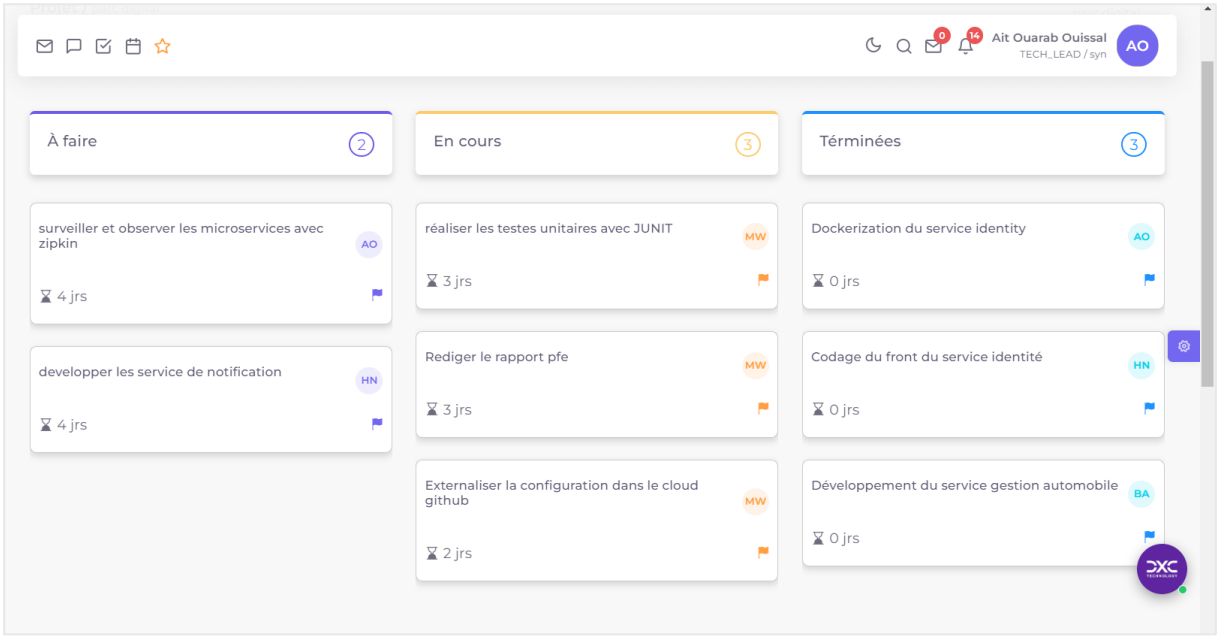


Figure 83: Suivie des tâches par le chef d'équipe 1/2

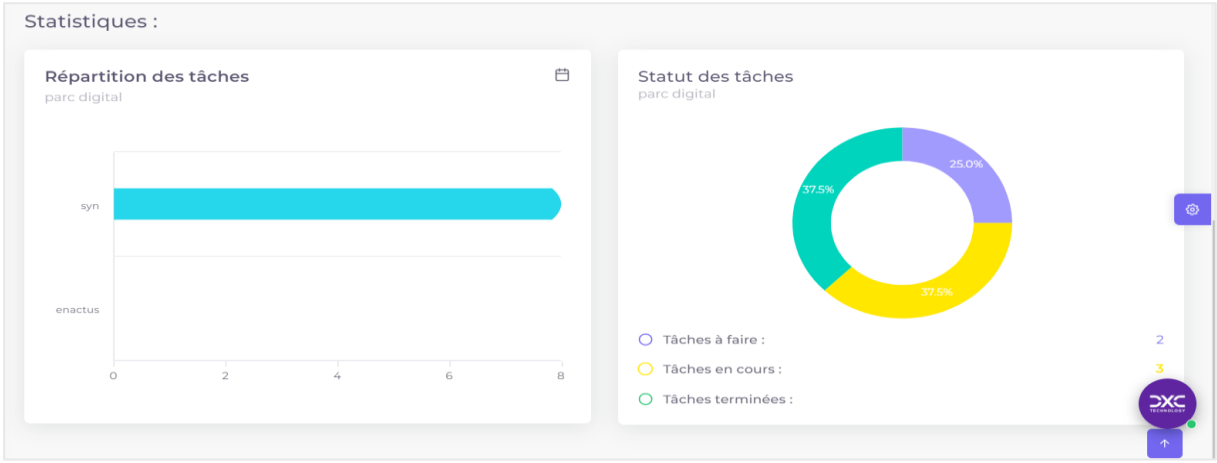


Figure 84: Suivie des tâches par le chef d'équipe 2/2

I.2 Scénario : Ajout d'un nouveau collaborateur

➢ **Accès à l'espace « Mon département » :** Après authentification, notre manager KHOUNA Younes peut accéder à l'espace « Mon département » afin de consulter l'équipe et y ajouter le collaborateur concerné.

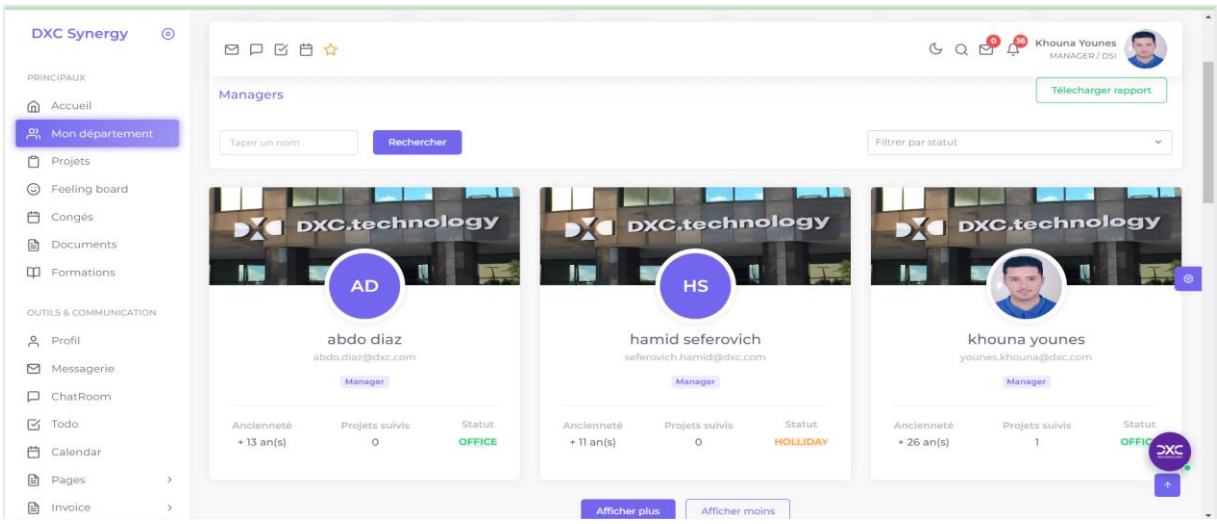


Figure 85: Page espace "mon département" 1/3

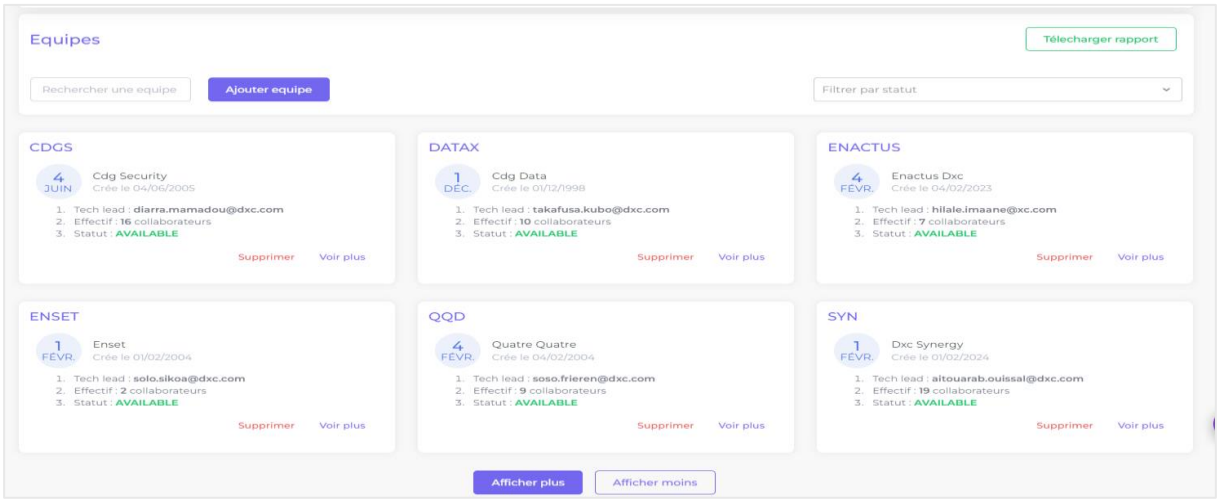


Figure 86: Page espace "mon département" 2/3

Figure 87: Page espace "mon département" 3/3

➢ **Ajout du nouveau collaborateur :** Après avoir sélectionné l'équipe **DXC Synergy** pour accueillir le nouveau collaborateur, il est désormais possible de procéder à son ajout.

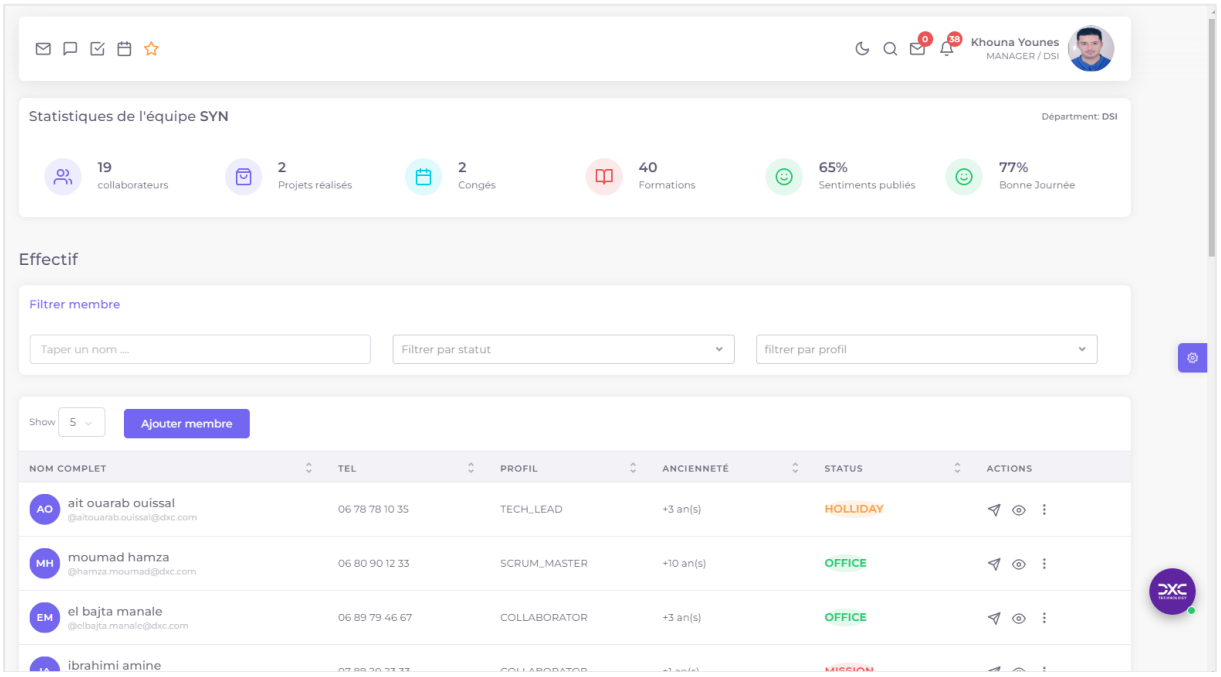


Figure 88: Espace équipe « DXC Synergy » 1/3

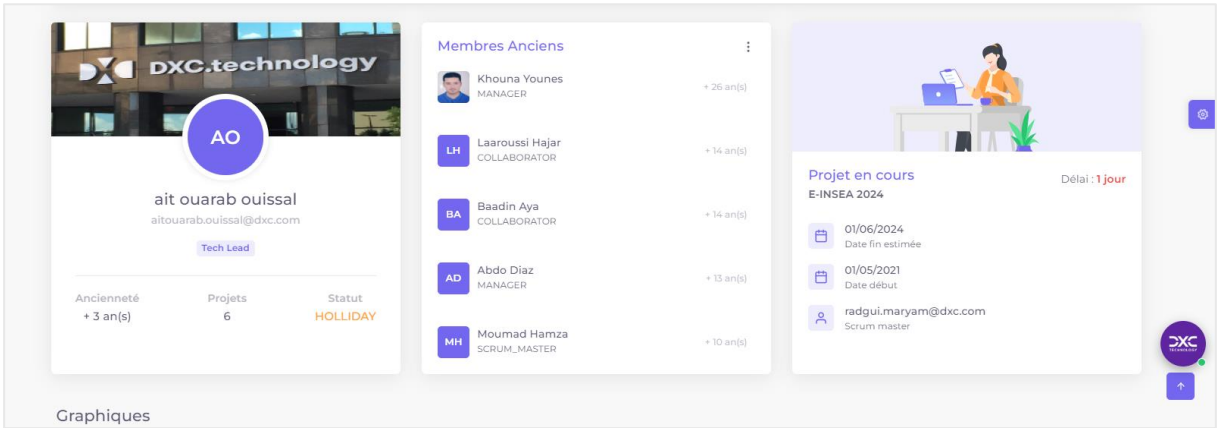


Figure 89: Espace équipe « DXC Synergy » 2/3

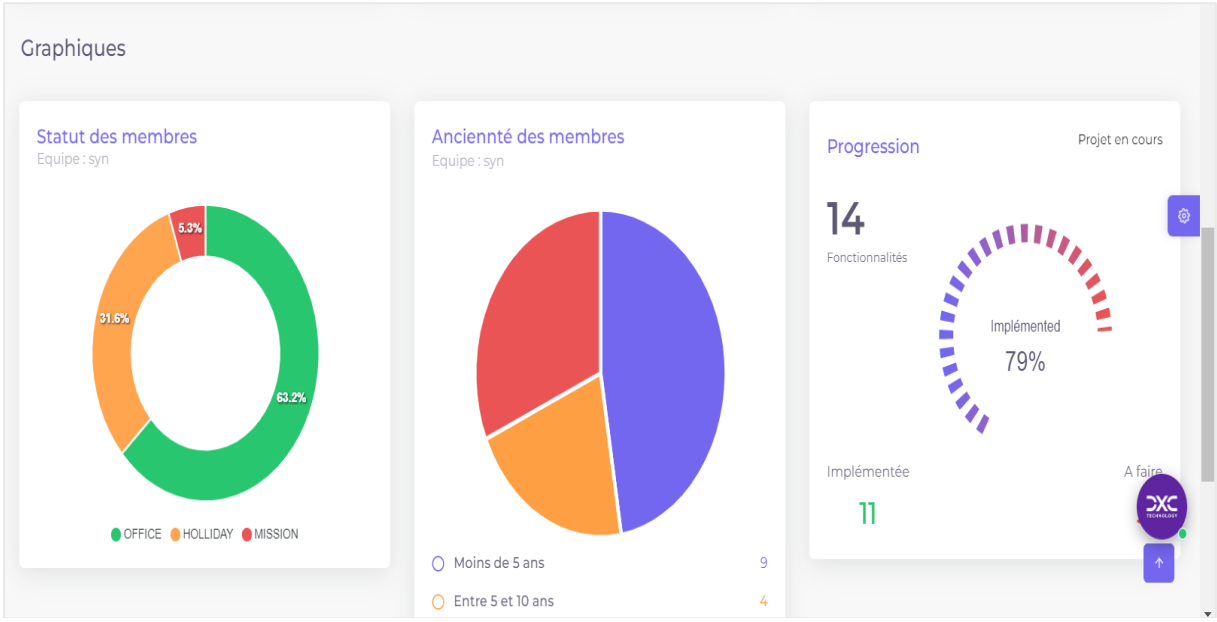
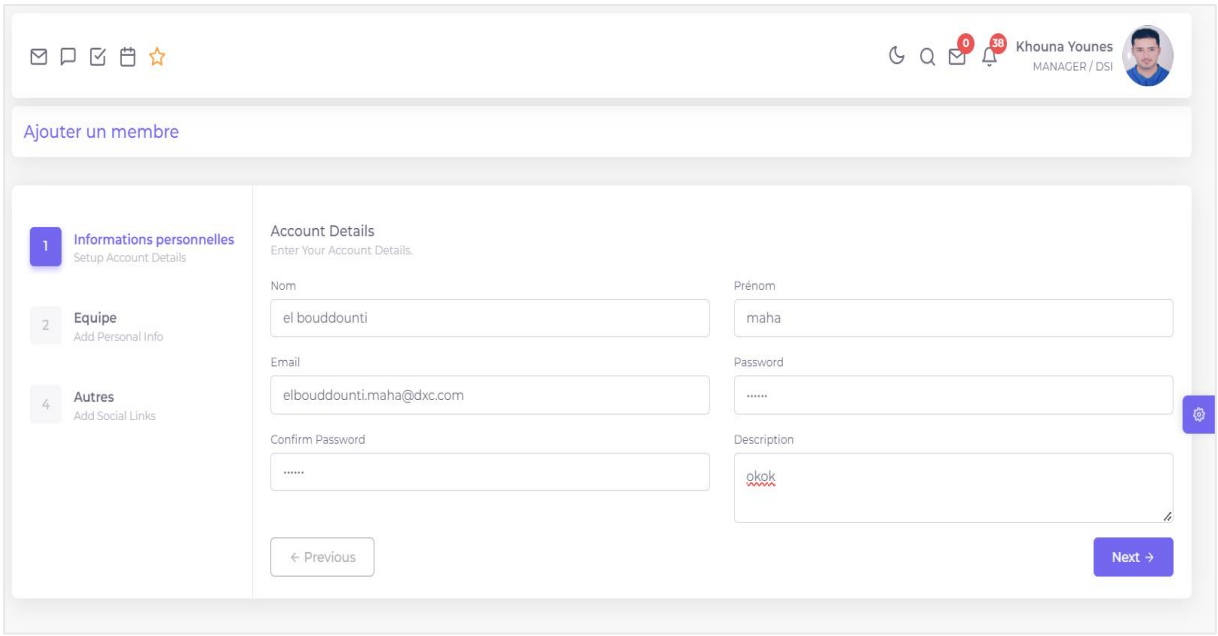
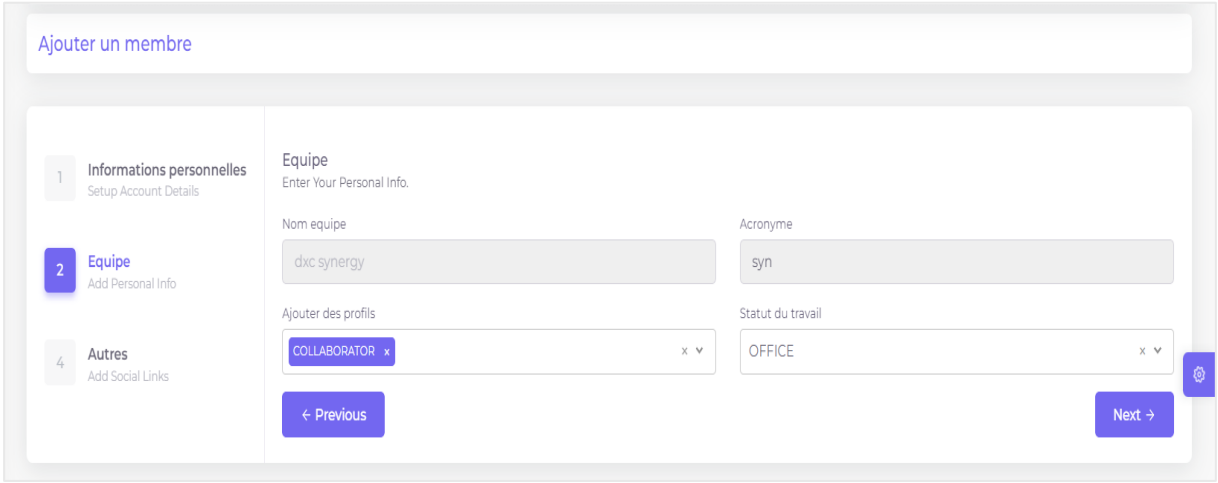


Figure 90: Espace équipe « DXC Synergy » 3/3

Cliquons sur le bouton ajouter et remplissons le formulaire d'ajout du collaborateur.





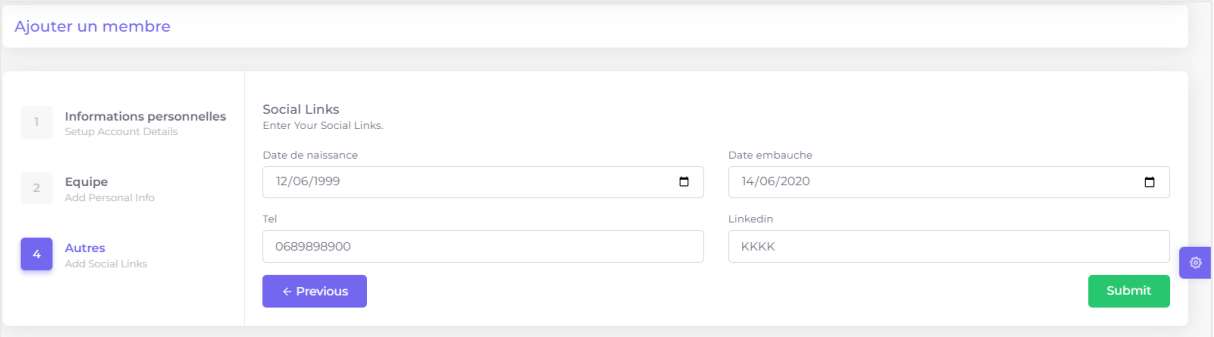


Figure 91: Formulaire ajout du nouveau collaborateur

Après un clique le bouton sur « ajouter », notre nouvelle collaboratrice **Mme ElBOUDDOUNTI Maha** est ajoutée à l'équipe avec succès, nous pouvons ainsi la **rechercher** dans la barre de recherche en saisissant son nom.

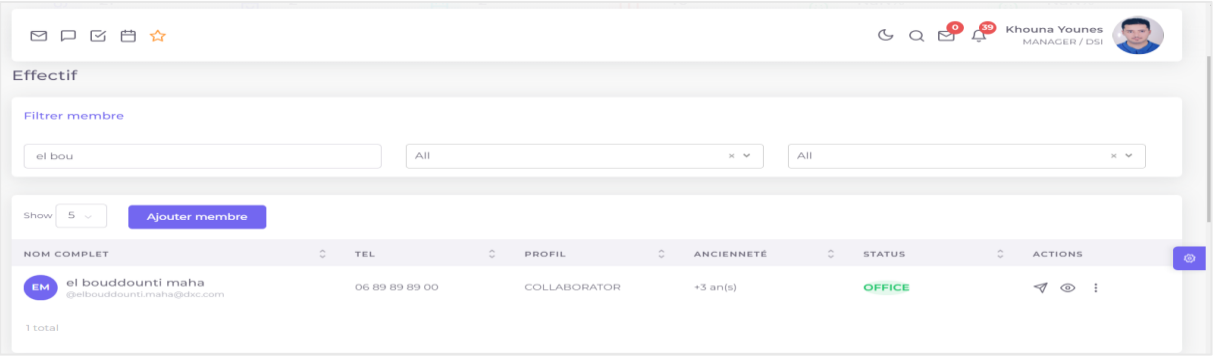


Figure 92: Ajout du nouveau collaborateur avec succès

➢ **Notification du chef d'équipe :** Après l'ajout d'un nouveau membre dans l'équipe, le chef d'équipe recevra une notification. (De même pour les membres et les managers)

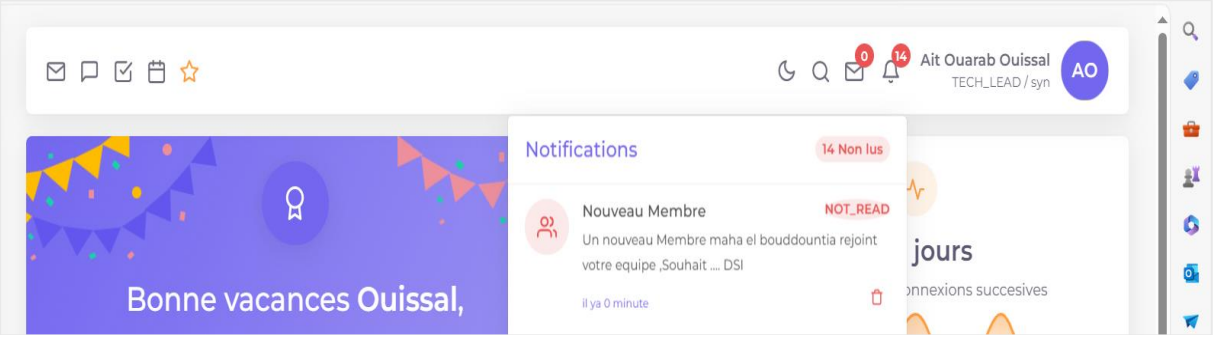


Figure 93: Notification du chef d'équipe "Nouveau membre"

➢ **Authentification du nouveau membre :** Le nouveau membre peut désormais se connecter à la plateforme. Sa **première notification** sera un message de bienvenue de l'application. **(Elle préfère le mode sombre)**

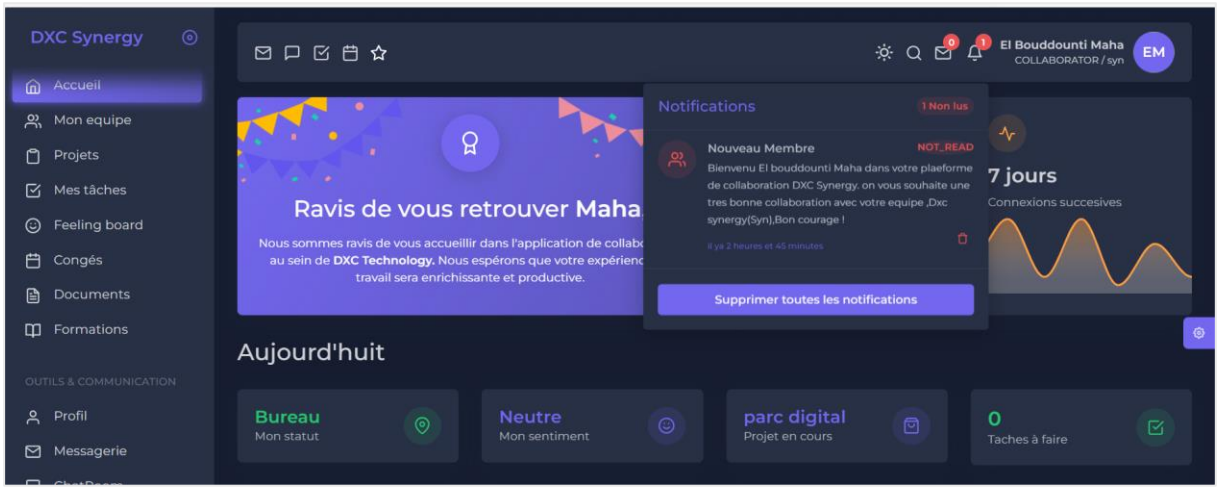


Figure 94: Authentification du nouveau collaborateur

➢ **Marquer la notification comme lue :** Le nouveau membre peut ainsi marquer la notification comme lue.

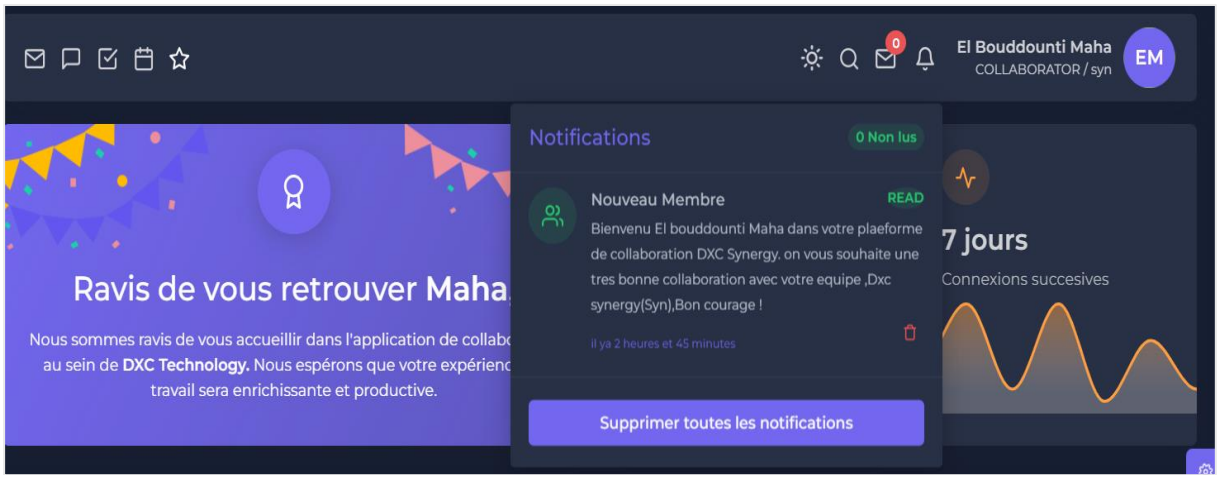


Figure 95: Marquer la notification de bienvenue comme lue

I.3 Scénario : Traitement d'une demande de congé

➢ **Envoie de la demande de congé :** Dans ce scénario, la collaboratrice **HOUARI Noura** dispose d'un solde suffisant pour envoyer une demande de congé (en indiquant son remplaçant) au manager **KHOUNA Younes**, qui la traitera.

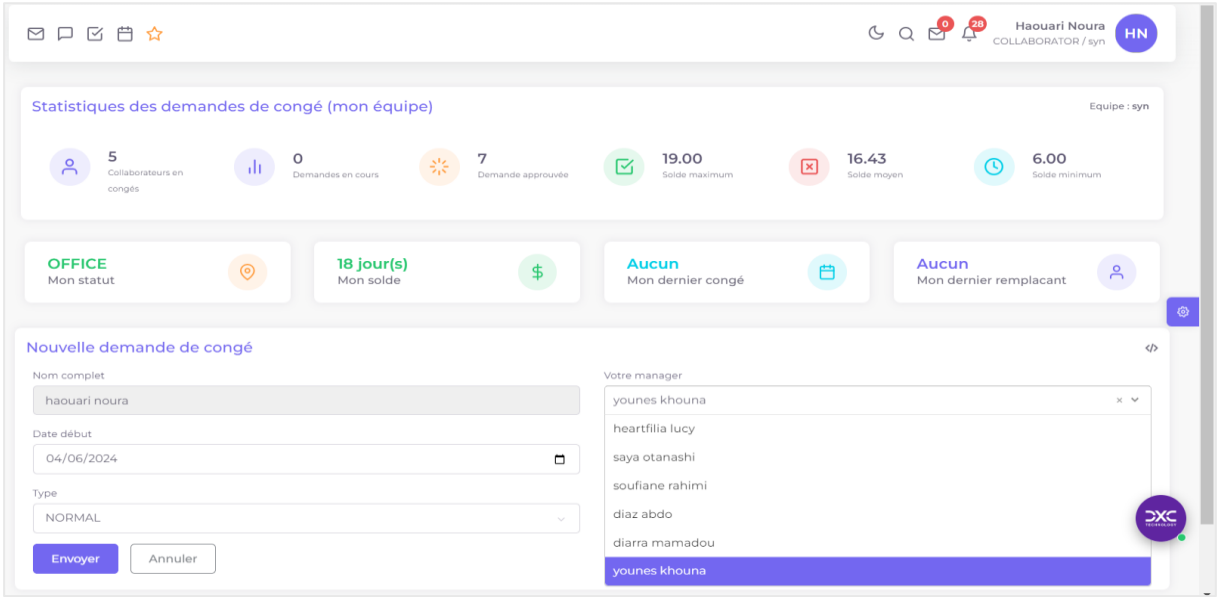


Figure 96: Formulaire demande de congé

La demande est ainsi envoyée au manager avec succès.

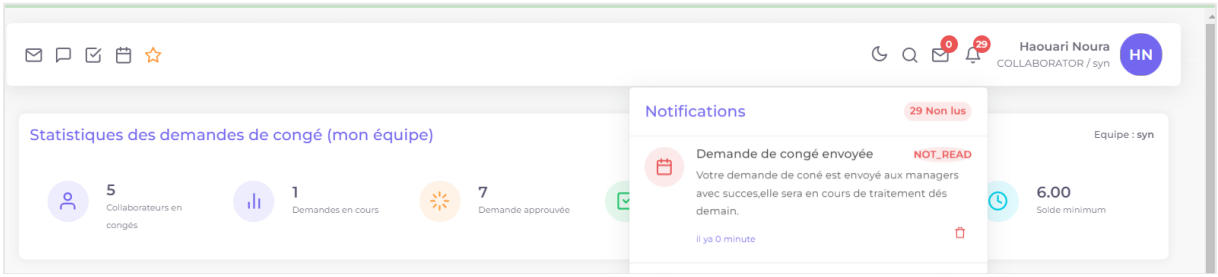


Figure 97: demande de congé envoyée avec succès

NB : l'utilisation du nom et prénom (ou encore email) **autre que le remplaçant** s'explique par le fait que dans le kafka listener ,nous avons affecté tous les adresses email des utilisateurs au notre à savoir « younes.khouna@gmail.com » pour éviter d'envoyer des courriers aux anonymes. (Voir configuration du kafka listener figure 124).

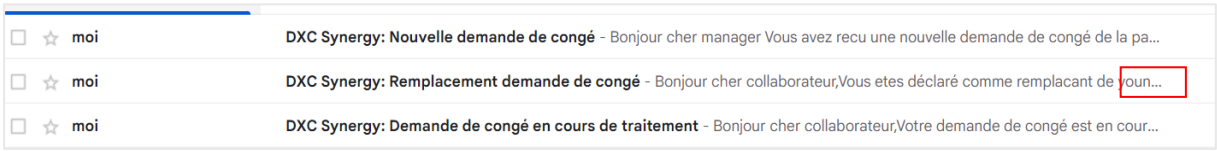


Figure 98: notification du remplaçant, manager et demandeur via SMTP

Figure 99: Progression de la demande de congé

➢ **Réception et traitement de la demande :** La demande de congé envoyée par la collaboratrice **Mme. HAOUARI Noura** est maintenant reçue par son manager, qui peut ainsi **approuver** ou rejeter cette dernière.

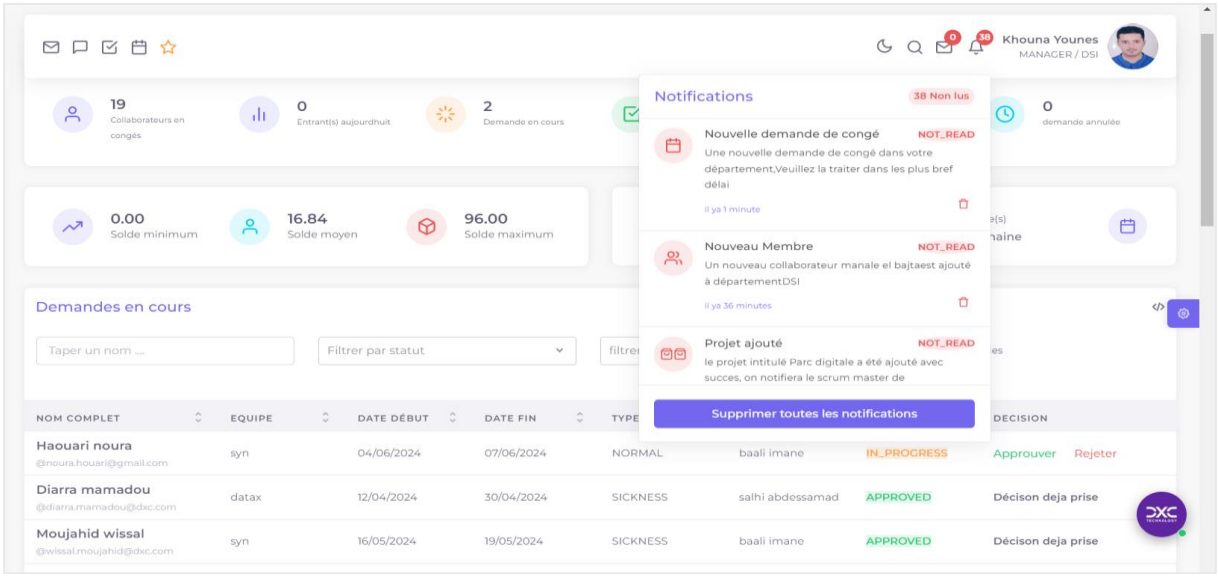


Figure 100: Réception et traitement de la demande de congé

➢ **Notification de l'approbation de la demande :** Une fois que le manager a approuvé la demande de congé, une notification en temps réel sera envoyée dans l'application ainsi que via le protocole SMTP au demandeur.

Figure 101: notification du traitement de la demande de congé

Figure 102: Mise à jour de la progression de la demande de congé

I.4 Scénario : Demander l'aide au Reference chatbot.

➢ **cliquer sur la bulle du référence chatbot :** : Dans ce scénario, le collaborateur (ou encore le manager) **M. KHOUNA Younes** peut demander l'aide au chatbot à propos des clauses de sécurité de l'organisation avec un simple clic.

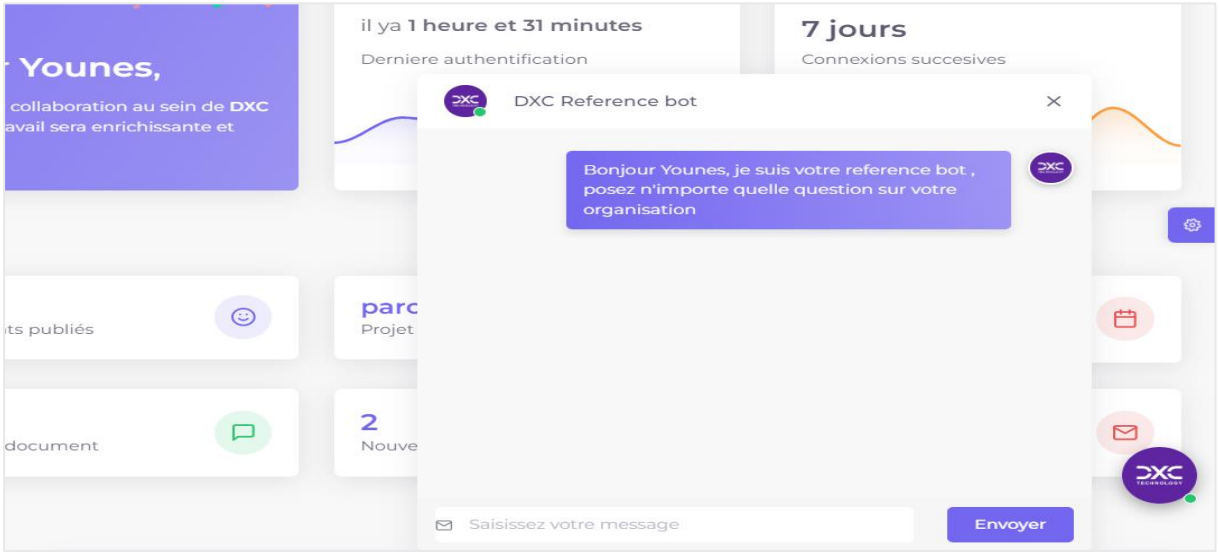


Figure 103: Ouvrir la bulle de conversation chatbot

➢ **Poser une question :** : Maintenant, Younes peut poser une question à savoir « quels sont mes obligations en termes de sécurité envers l'organisation ?»

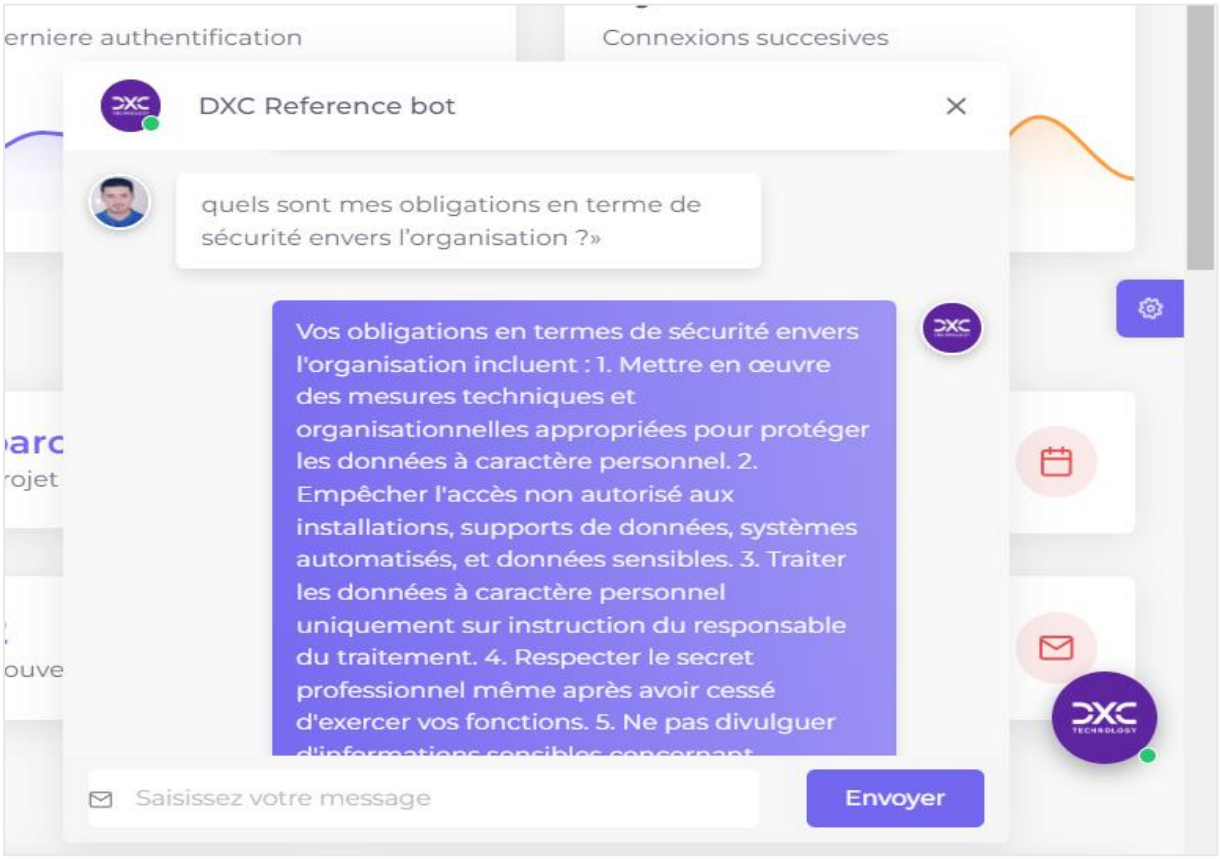


Figure 104: poser une question au Reference chatbot

I.5 Scénario : Changer sa photo de profil dans le forum (Développé par M. IBRAHIMI Amine)

➢ **Consulter son profil :** Une fois authentifié, notre manager peut accéder au forum et publier des activités, voir son profil ou encore commenter d'autres activités.

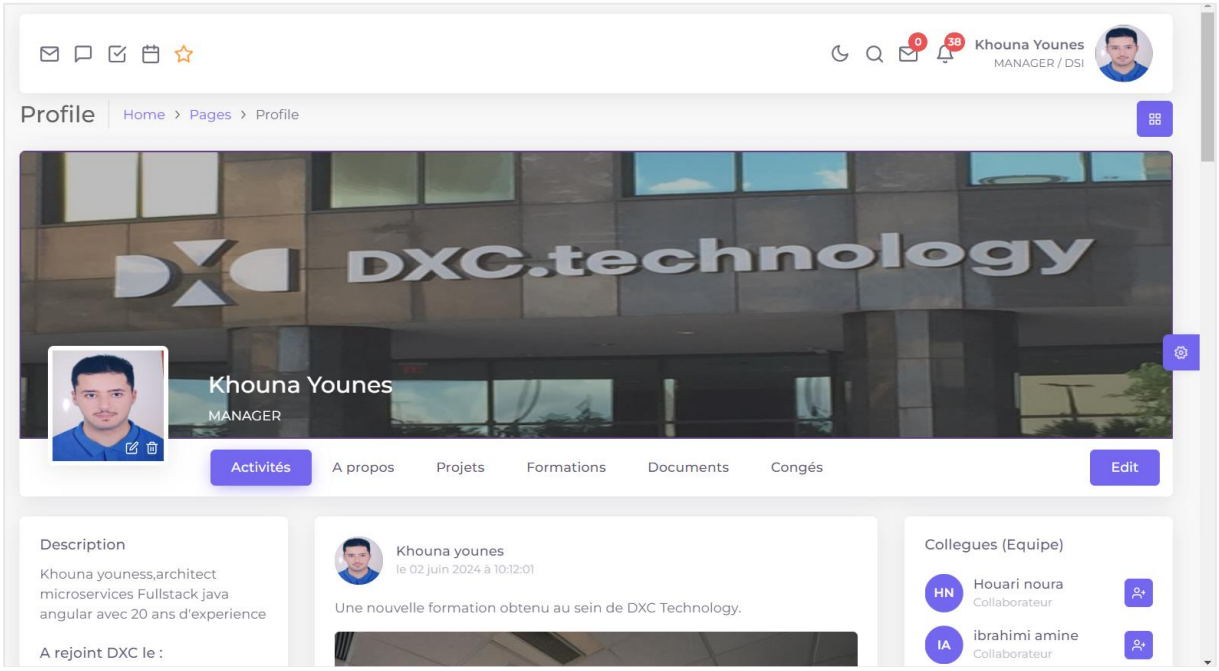


Figure 105: Forum de partage 1/2

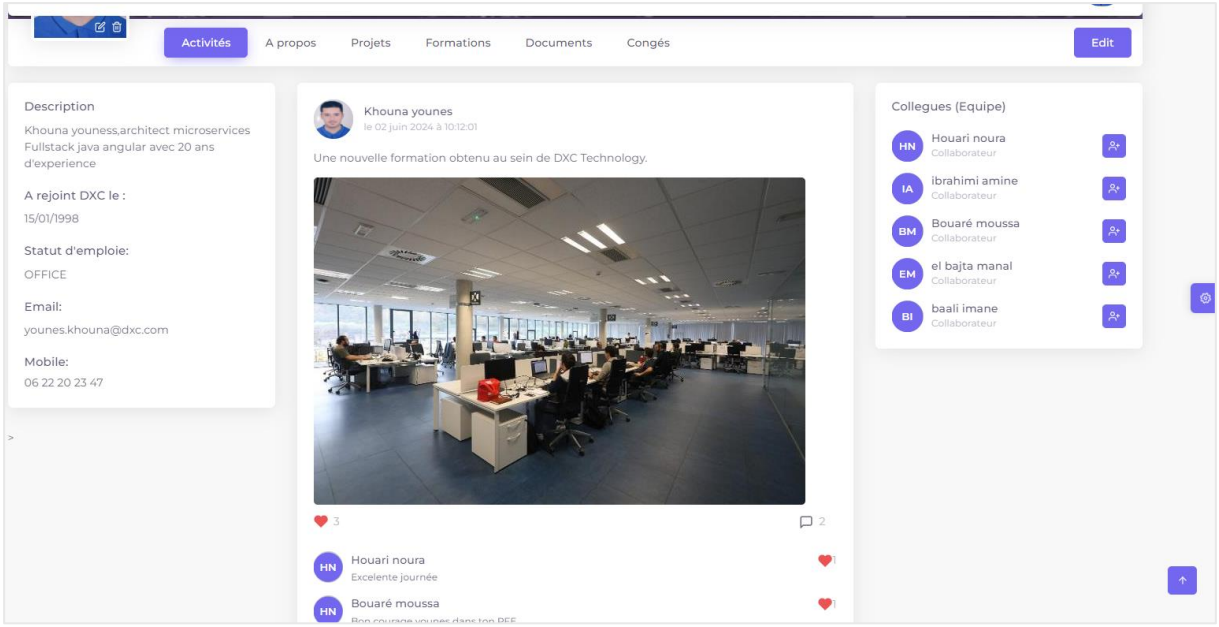


Figure 106: Forum de partage 2/2

➢ **Changer sa photo de profil :** notre manager peut changer sa photo de profil par un simple clic sur l'icône modifier.

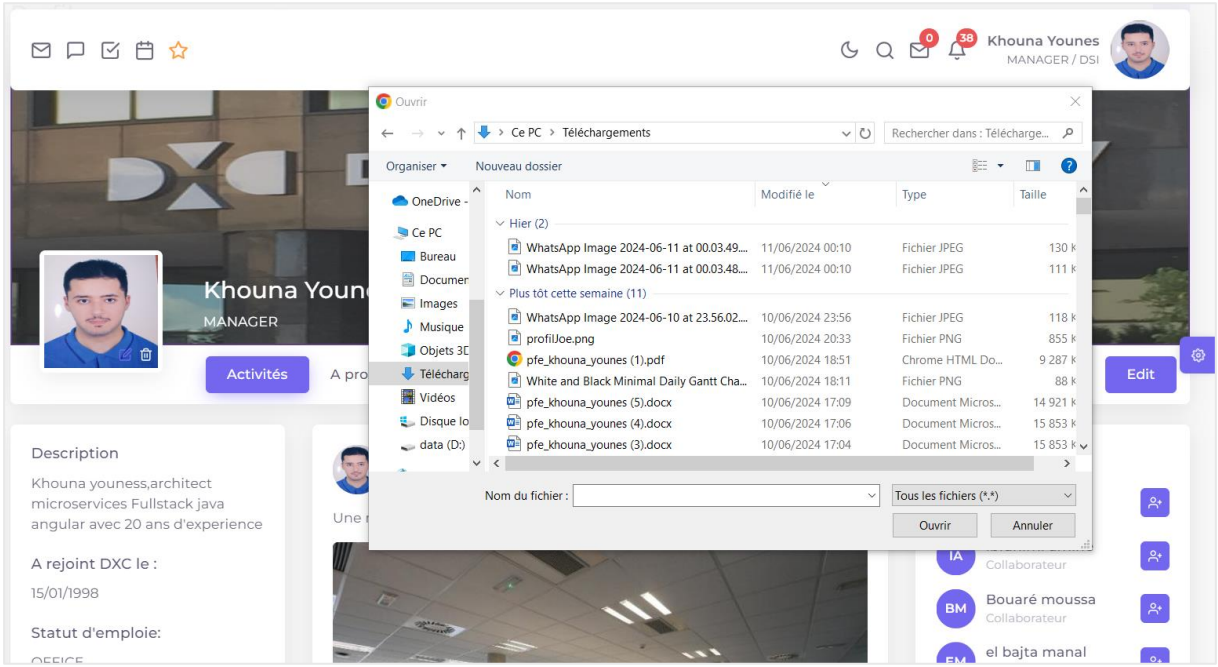


Figure 107: Changement de la photo de profil 1/2

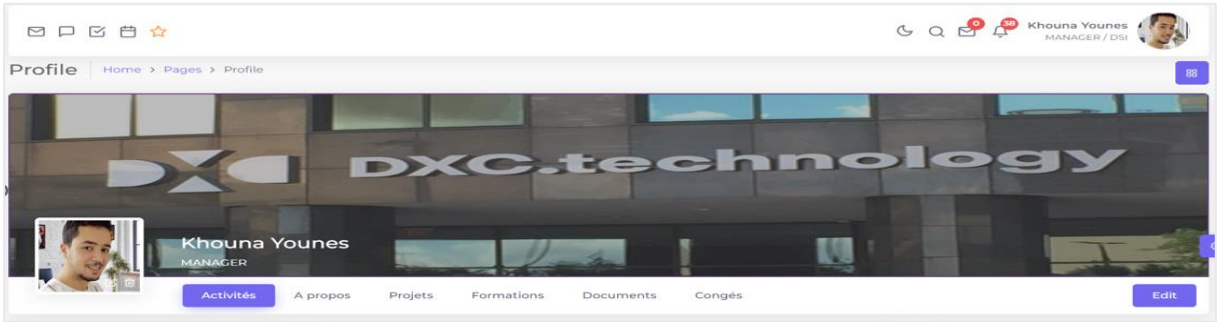


Figure 108: Changement de la photo de profil 2/2

I.6 Scénario : Envoyer un message privé (Développé par M. IBRAHIMI Amine)

➢ **Choisir une conversation :** Après avoir accédé à « l'espace messagerie », le manager KHOUNA Younes peut envoyer un message à un contact à savoir **Mme. IBRAHIMI Safae**.

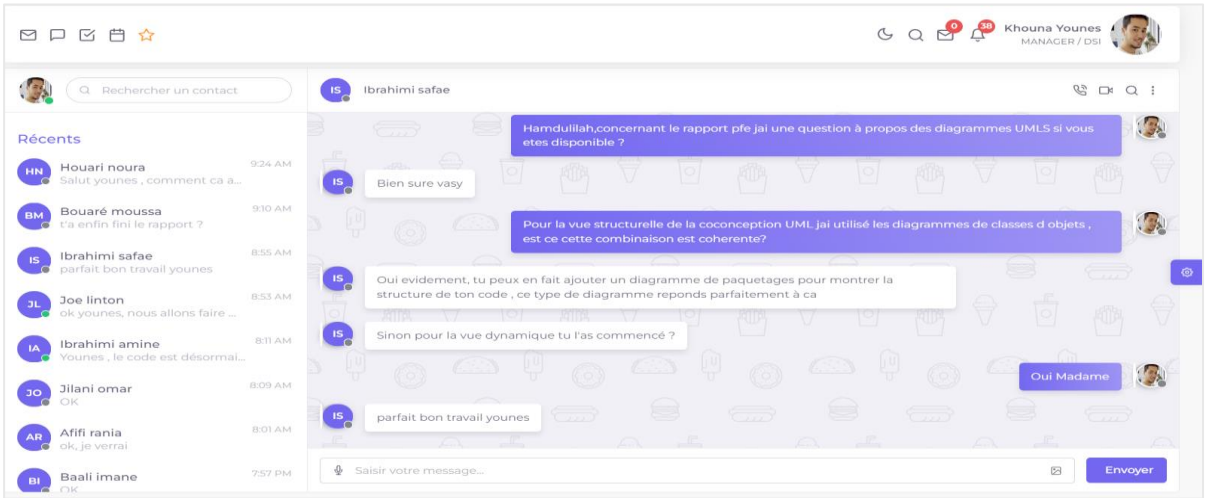


Figure 109: Choisir une conversation

➢ **Saisir un nouveau message :** Maintenant le manager peut ainsi envoyer un nouveau message.

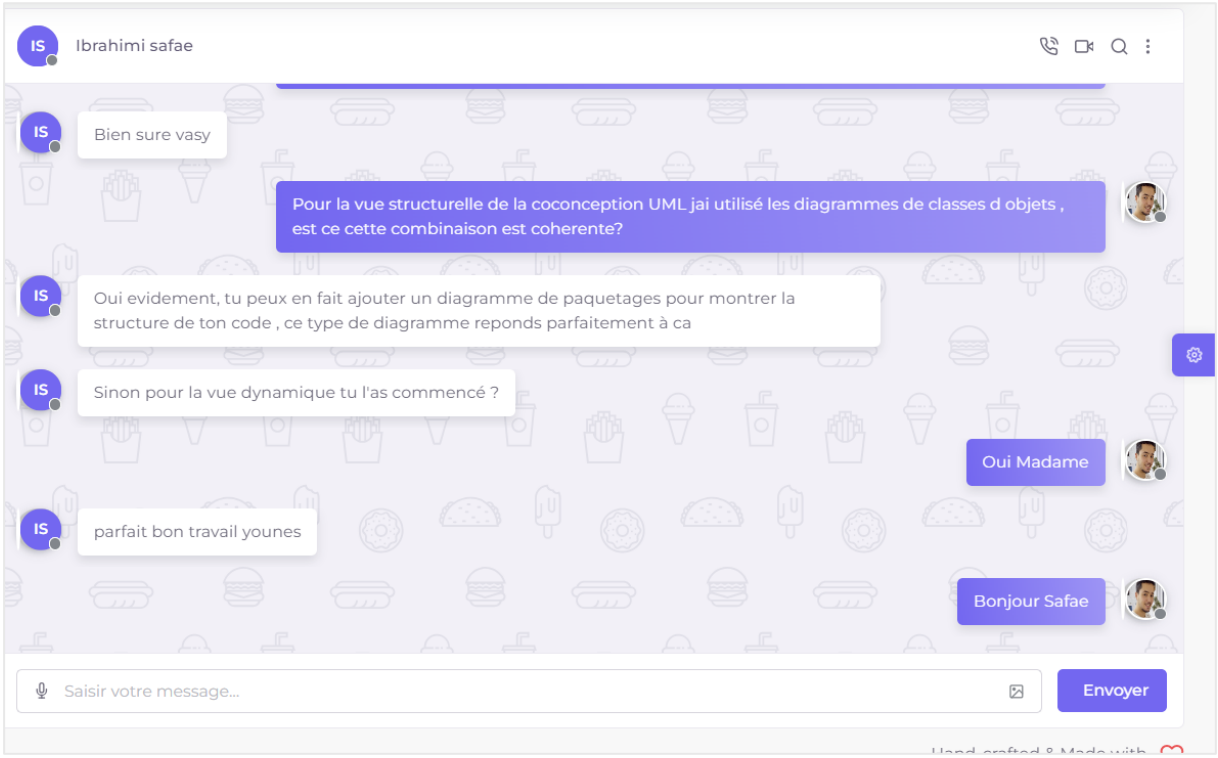


Figure 110: Envoyer un nouveau message

I.7 Scénario : Publier son sentiment dans le feeling board

➢ **Liste des sentiments actuels :** : Ici, nous affichons le tableau des sentiments côté manager avant que la collaboratrice **Mme EL BOUDDOUNTI** Maha ne publie son sentiment du jour (**le jeudi 06/06/2024**).

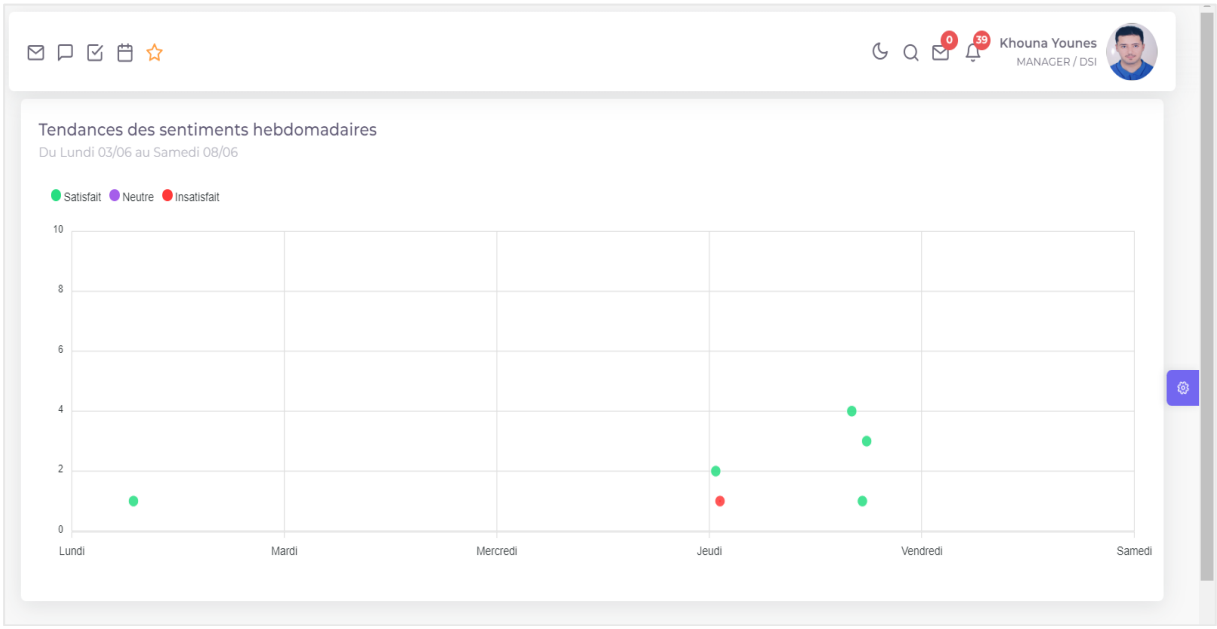


Figure 111: Feeling board des sentiments initiaux

➢ **Publier son sentiment :** : Notre collaboratrice, après avoir accédé à l'espace « Feeling board », peut ainsi publier son sentiment.

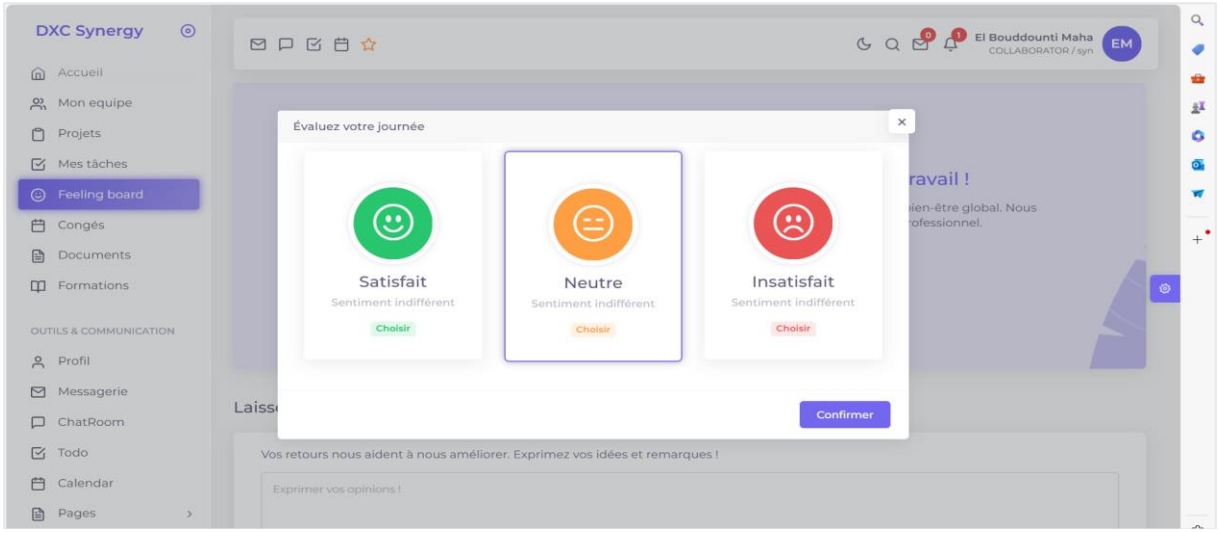


Figure 112: publier son sentiment dans le feeling board1/2

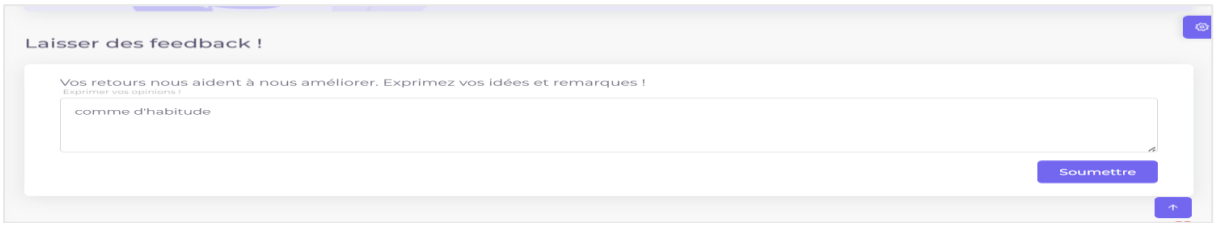


Figure 113: publier son sentiment dans le feeling board 2/2

➢ **Mise à jour du feeling board (Manager) :** : Une fois le nouveau sentiment de notre collaboratrice est publié, il sera automatiquement inséré dans le feeling board. **(C'est 21h à Rabat le mode sombre est automatiquement appliqué)**

Figure 114 : Mise à jour du feeling board (mode sombre)

I.8 Scénario : Télécharger un document vue manager (Développé par M. IBRAHIMI Amine)

➢ **Accès à l'espace « Documents » :** Cet espace est dédié au patrimoine documentaire de l'organisation. Un manager peut consulter tous les documents publiés et en téléverser de nouveaux.

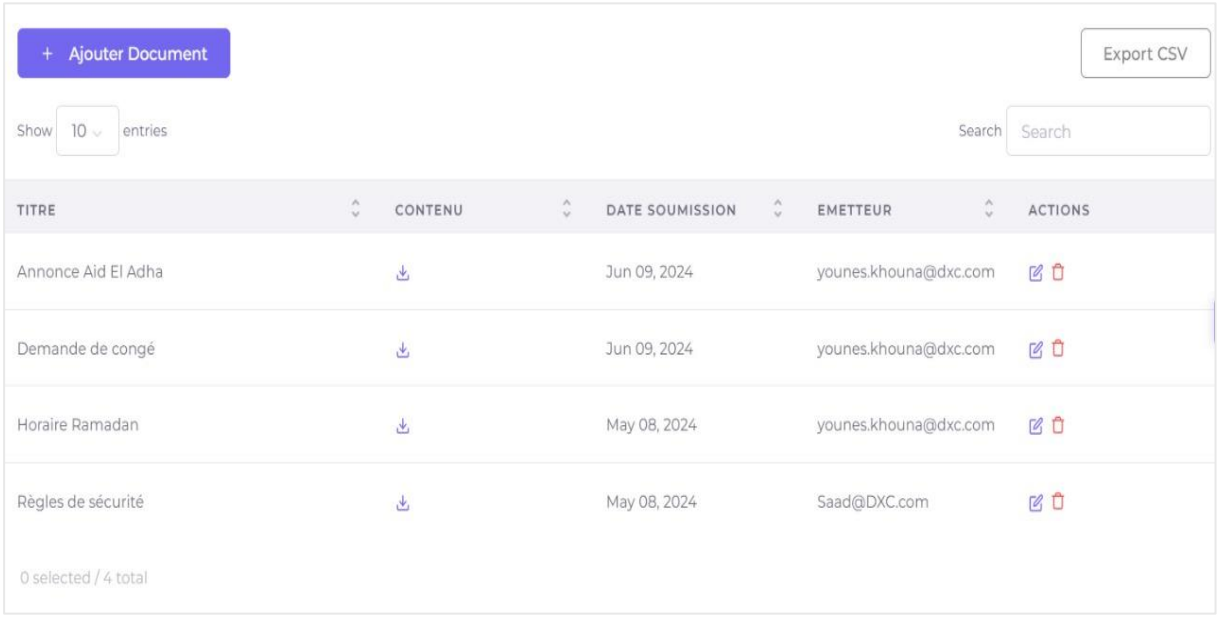


Figure 115: Liste des documents publiés dans l'espace document

➢ **Télécharger le document :** Notre manager M. KHOUNA Younes va télécharger le document intitulé « Règles de sécurité » avec un simple clic sur l'icône « Télécharger ».

NB : Pour la **confidentialité** des données de l'organisation, nous avons modifier le contenu du fichier des règles de sécurité par **la liste des utilisateurs de la plateforme**.

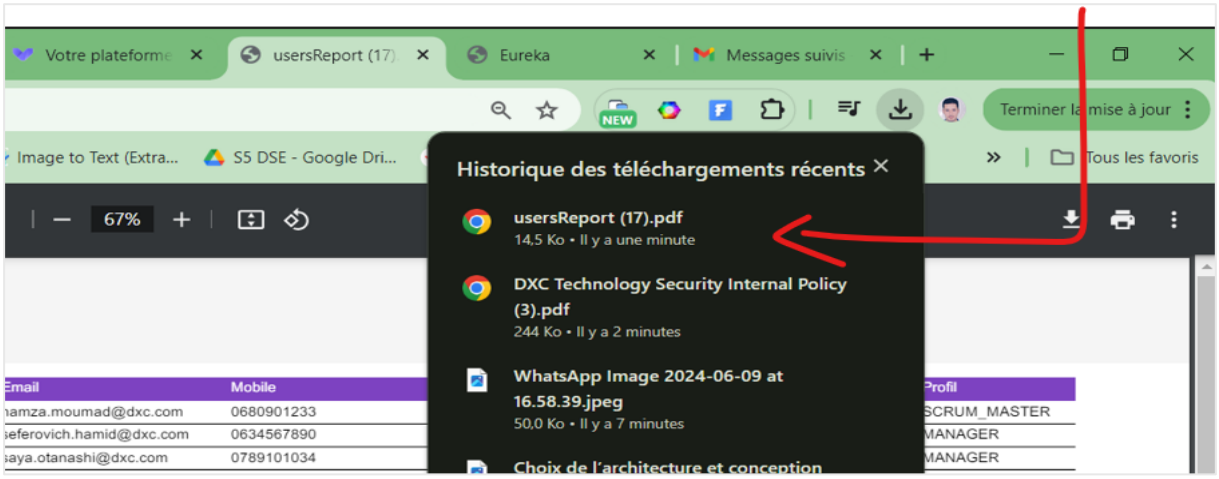


Figure 116: Téléchargement du document

II. Point de vue développeur

Cet optique nous permettra de concrétiser l'architecture logicielle et les exigences techniques du projet mentionnées lors de la phase de spécification fonctionnelle.

II.1 Codage des microservices

Comme illustré dans la phase de conception, notamment à travers les diagrammes de paquetages, la figure (117) illustre la structure du code de notre projet. Cette organisation permet une **évolutivité facile**, garantissant que notre système pourra s'adapter aisément aux futures exigences et modifications.

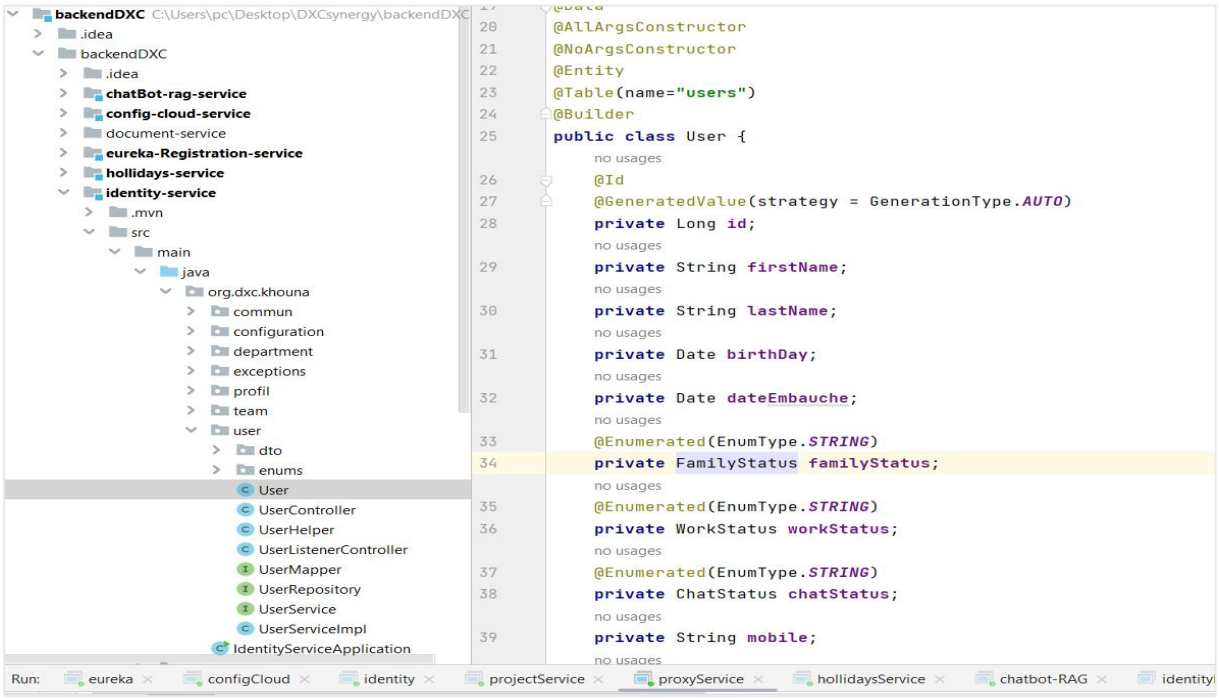


Figure 117: Structure du code

II.2 Externalisation de la configuration.

Nous avons centralisé la configuration de nos microservices dans le cloud, plus précisément dans GitHub. Cette approche permet de regrouper toute la configuration commune entre les microservices dans un fichier nommé « application.properties ». Les configurations spécifiques à chaque microservice sont stockées dans des fichiers distincts, nommés selon le format « nom_du_microservice.properties ». Cela facilite la gestion et **la maintenance des configurations,** assurant une cohérence et une mise à jour simplifiée pour l'ensemble du système.

La figure (118) montre cette approche de centralisation de la configuration.

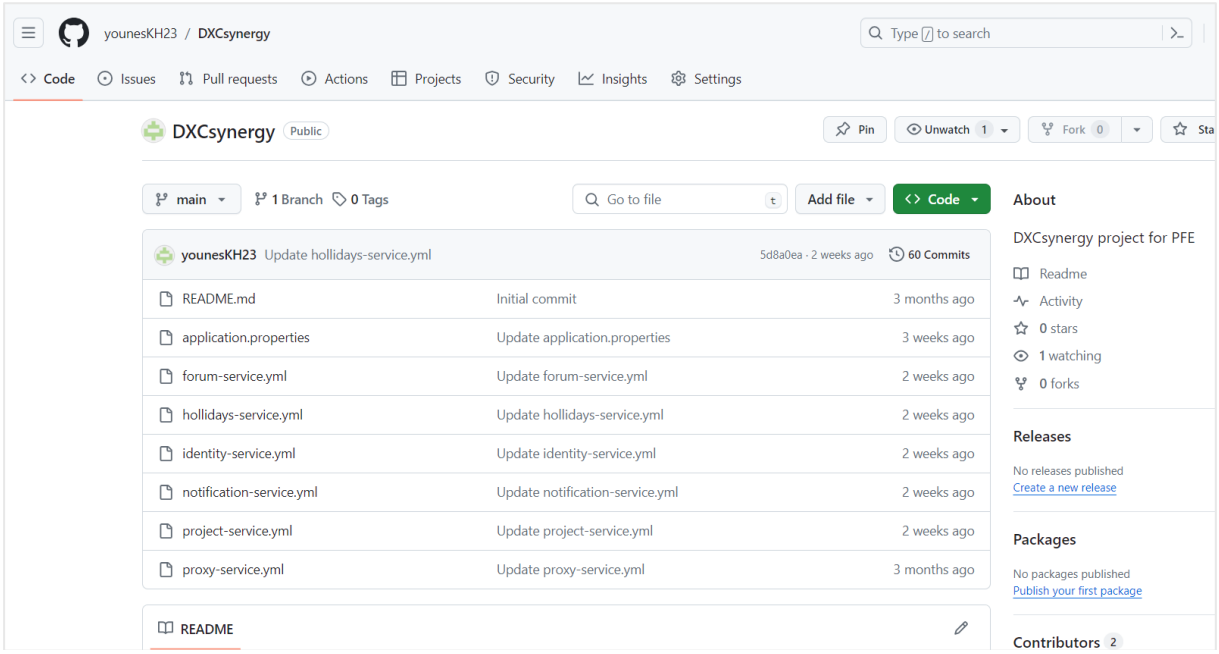


Figure 118: Externalisation de la configuration des microservices

II.3 Inscription des microservices et équilibrage de charge.

Grâce au service « eurekaRegistration-service », Eureka prend en charge l'équilibrage de charges des microservices. Cela garantit une forte résilience et une scalabilité horizontale pour notre application.

La figure (119) illustre un exemple **d'équilibrage de charge** avec Eureka en fournissant **deux instances** pour le microservice **« projectScrum-service »**. Cela permet de répartir les demandes entre plusieurs instances, assurant ainsi une meilleure performance et une **disponibilité** accrue de ces services.

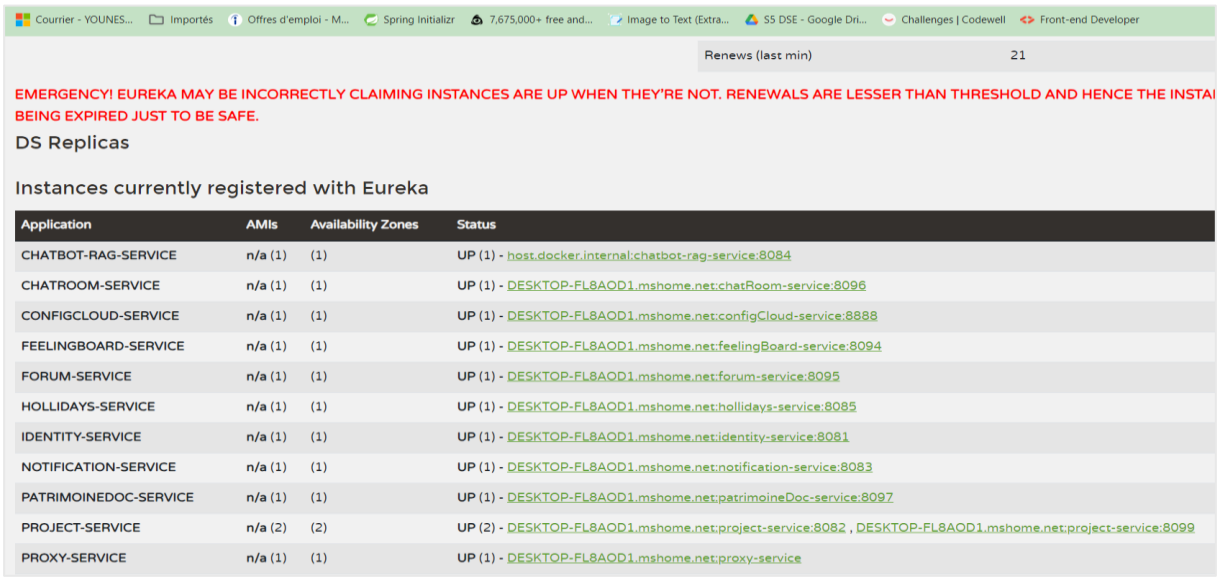


Figure 119: Equilibrage de charge et résilience grâce à Eureka

II.4 Documentation

Pour la documentation, nous avons pris l'exemple du microservice identity-service. Cette documentation permet de visualiser tous les détails du codage, tels que les entités, les classes DTO, les contrôleurs, et les API. Grâce à cette documentation, les développeurs peuvent comprendre rapidement et facilement la structure et le fonctionnement du microservice.

La figure (120) montre un aperçu de la documentation du microservice « identity-service ».

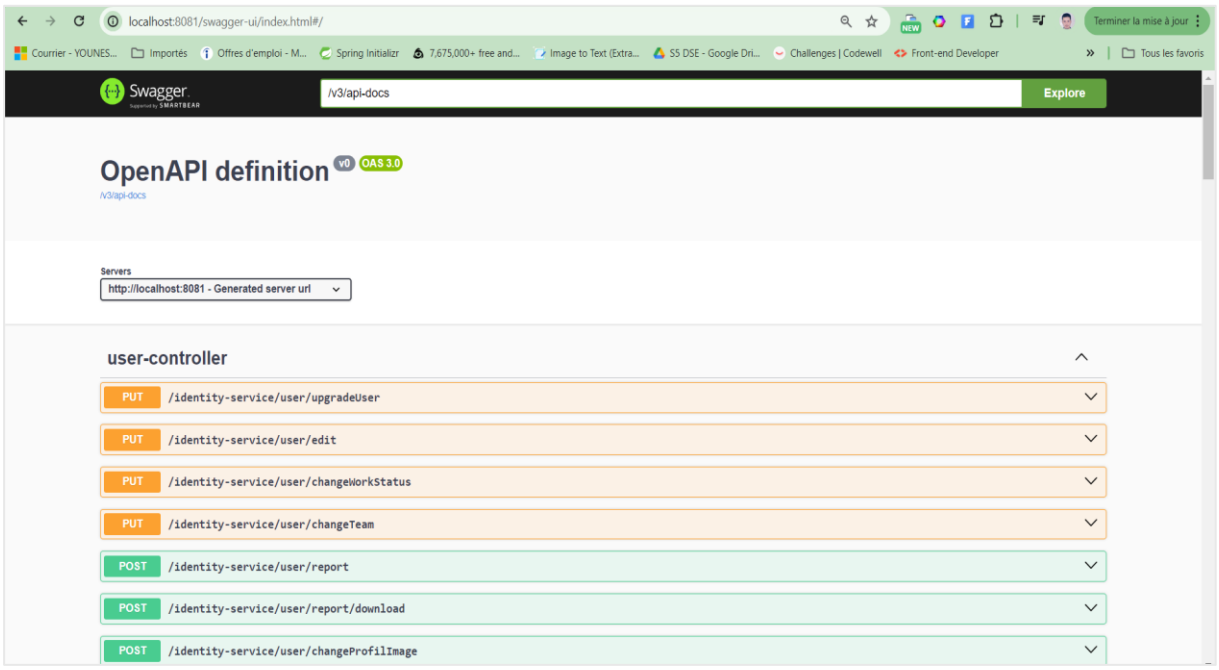


Figure 120: Documentation de <<identity-service>>

II.5 Implémentation du proxy-service

Pour cette phase, nous avons illustré un exemple concret de routage des microservice comme indiqué dans la figure (121).

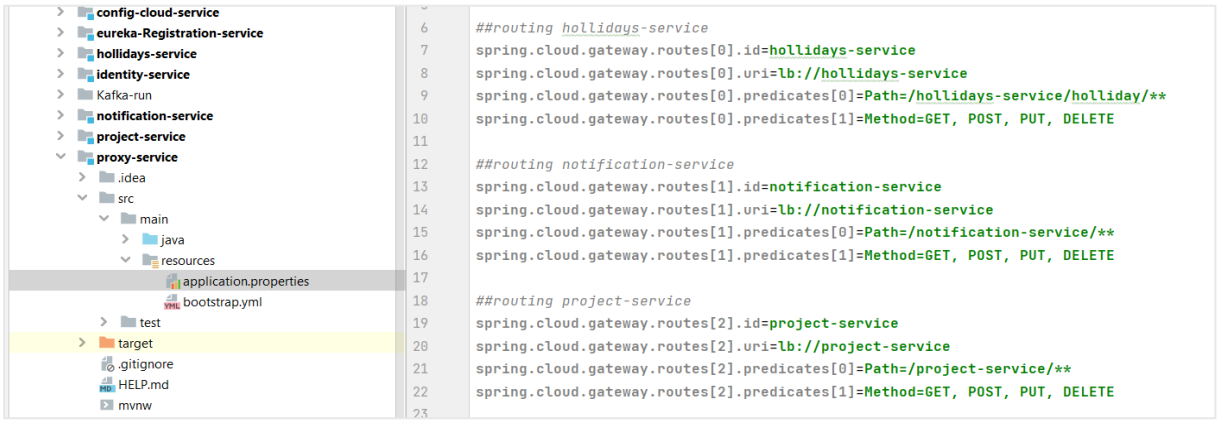


Figure 121: Routage des microservice

II.6 Traçabilité des requêtes client http.

Ici nous avons suivi le chemin exact de chaque requête http provenant de l'utilisateur de l'application.

Par exemple, une authentification a passé par le **« proxy-service »** et puis par le **« identity-service ».** La figure (122) illustre la traçabilité de toutes les requêtes effectuées pour un moment donnée.

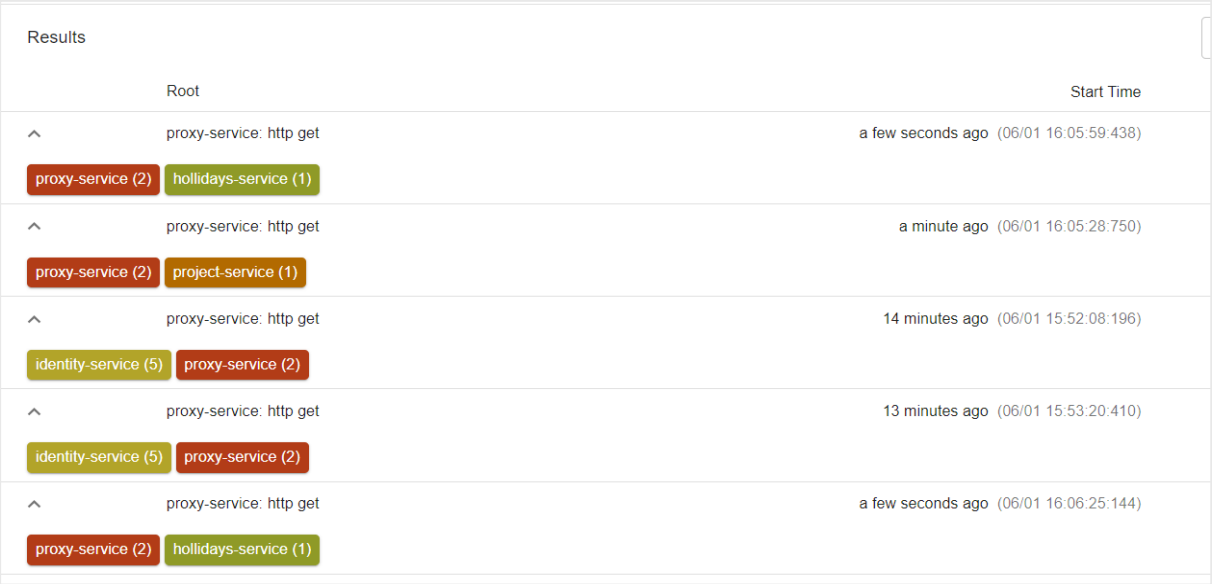


Figure 122: Traçabilité des requêtes HTTP

II.7 Sécurité et confidentialité des données.

Nous avons sécurisé les données sensibles des utilisateurs de la plateforme via un algorithme de hachage irréversible à savoir **l'algorithme Bcrypt**.

La figure (123) illustre le hachage des mots de passes des utilisateurs.

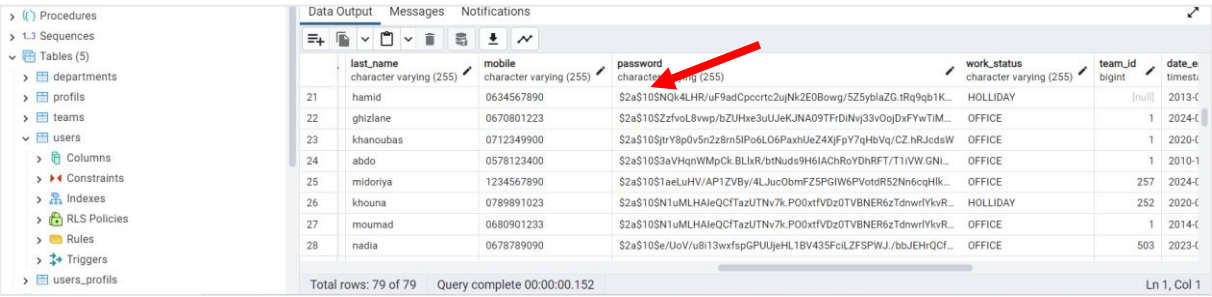


Figure 123: Sécurité des données avec l'algorithme Bcrypt

II.8 Configuration du kafka Listener

Nous avons configuré le Kafka Listener principalement dans le service de notifications « notification-service », qui écoute les événements provenant des autres services métiers.

La figure (124) illustre l'événement lié à une nouvelle demande de congé.

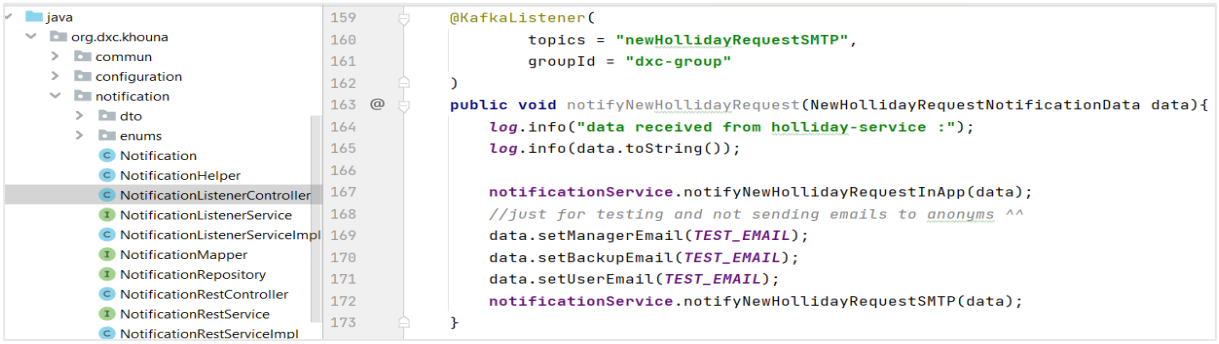


Figure 124: Configuration du kafka Listener

II.9 Implémentation du chatbot RAG

Nous avons mis en œuvre les phases opérationnelles de notre bot de référence en utilisant l'approche RAG, conformément à l'architecture décrite précédemment (voir figure 71). Cela comprend la phase d'**ingestion** jusqu'à son **utilisation par l'utilisateur final.**

La figure (125) illustre l'implémentation du chatbot RAG plus précisément la couche service.

Figure 125: Implémentation du chatbot RAG

III. Synthèse

Dans ce chapitre, nous avons exploré la réalisation du projet selon deux perspectives distinctes : celle du développeur technique et celle de l'utilisateur fonctionnel. Nous avons détaillé les différents scénarios d'utilisation de l'application pour chaque module spécifié, en abordant les aspects techniques de développement ainsi que les fonctionnalités et l'expérience utilisateur.

Cette approche nous a permis de mieux comprendre les exigences et les interactions entre les composants techniques et les besoins des utilisateurs finaux.

Conclusion

Dans ce dernier chapitre, nous allons d'abord procéder à une évaluation complète du bilan du projet, en analysant les objectifs atteints, les défis rencontrés et les leçons apprises. Ensuite, nous aborderons les perspectives, en discutant des améliorations possibles, des opportunités de développement et des orientations futures pour garantir la pérennité et le succès continu du projet.

I. Bilan

Le bilan de notre stage chez DXC Technology Maroc était extrêmement positif. D'un point de vue méthodologique, l'adoption des méthodes agiles, telles que SCRUM, a joué un rôle crucial dans la réalisation des besoins fonctionnels et non fonctionnels requis dans le cahier des charges. Cette approche a permis une meilleure organisation, une flexibilité accrue et une collaboration efficace au sein de l'équipe.

Sur le plan technique, nous avons eu l'opportunité d'apprendre de nouvelles technologies, notamment Angular pour le développement frontend et le Framework Spring AI pour l'intégration de l'intelligence artificielle dans notre projet. Nous avons également exploité la robustesse des modèles de langage large (LLM) et de l'intelligence générative dans notre projet, ce qui a permis de créer des solutions innovantes et d'améliorer significativement les fonctionnalités de l'application.

Enfin, d'un point de vue professionnel, ce stage nous a permis de découvrir le milieu professionnel et l'industrie en profondeur, nous offrant une précieuse expérience pratique et une meilleure compréhension des dynamiques du secteur.

II. Perspectives

Pour les perspectives, nous allons nous concentrer sur le déploiement de l'application sur le cloud avec AWS. Cette migration vers le cloud offrira de nombreux avantages, tels que l'évolutivité, la flexibilité, et une meilleure gestion des ressources, tout en assurant une disponibilité élevée et une sécurité renforcée.

En parallèle, nous prévoyons d'intégrer Spring Batch pour gérer les traitements par lots de manière efficace. Cette approche permettra d'améliorer la performance et la fiabilité des processus de traitement de données, répondant ainsi aux besoins croissants de l'application en matière de gestion et d'analyse de grandes quantités de données.

Ces initiatives contribueront à renforcer l'architecture de notre projet et à garantir sa pérennité et son efficacité à long terme.

Glossaire

CDG (sigle) : La Caisse de dépôt et de gestion du Maroc (CDG), créée en 1959, est une institution financière publique qui gère l'épargne à long terme au Maroc, notamment les fonds des réserves de la CNSS, de la Caisse d'épargne nationale, des fonds de consignations et des fonds provenant des professions juridiques. La CDG gère également la Caisse nationale de retraites et d'assurances (CNRA) qui administre le régime collectif d'allocation de retraite (RCAR). En tant qu'investisseur majeur, la CDG possède des filiales dans divers secteurs économiques. Khalid Safir en est le directeur depuis le 13 juillet 2022.

CRUD : Acronyme informatique anglais CRUD pour Create (créer), Read (lire), Update (mettre à jour), Delete (supprimer). CRUD désigne les quatre opérations de base pour la persistance des données, en particulier le stockage d'informations en base de données.

Feeling board : Dans le Framework Scrum, le "feeling board" est un outil permettant aux membres de l'équipe de partager et de visualiser leurs émotions et sentiments à propos du projet ou des sprints en cours. Il favorise la transparence et la communication ouverte, aidant à identifier et à aborder les préoccupations ou les sources de motivation au sein de l'équipe. En facilitant l'expression des ressentis individuels, le feeling board contribue à améliorer la cohésion de l'équipe et à ajuster les processus pour mieux répondre aux besoins émotionnels des membres.

Go_gc_duration : Cette métrique est importante car elle mesure combien de temps les pauses dues au **garbage collection** durent. Lorsqu'une application Java effectue un cycle de garbage collection, elle doit parfois arrêter l'exécution normale pour libérer de la mémoire. Ces pauses peuvent affecter les performances de l'application, en particulier si elles sont longues ou fréquentes.

SCRUM : Scrum est un cadre de gestion de projet agile utilisé principalement dans le développement de logiciels, qui favorise la collaboration, la responsabilisation et l'amélioration continue. Il se compose de trois rôles principaux : le Product Owner (qui gère le backlog du produit), le Scrum Master (qui facilite les processus Scrum et élimine les obstacles) et l'équipe de développement (qui crée les incréments du produit.

RAG : L'approche RAG (Retrieval-Augmented Generation) en intelligence artificielle générative combine des techniques de recherche d'informations et de génération de texte. Elle consiste à utiliser un modèle de récupération pour trouver des documents ou des passages pertinents à partir d'une vaste base de données et un modèle génératif pour élaborer des réponses basées sur ces informations récupérées, améliorant ainsi la précision et la pertinence des réponses fournies par l'IA.

SRP : Le principe SRP (Single Responsibility Principle) en développement logiciel stipule qu'une classe ou un module doit avoir une seule responsabilité ou raison de changer. Cela signifie qu'elle doit encapsuler une fonctionnalité cohérente et indépendante des autres parties du système, facilitant ainsi la maintenance, les tests et l'évolution du code en réduisant les interférences entre différentes fonctionnalités et en améliorant la clarté et la modularité du code.

Time-To-Market : C’est un terme qui désigne la durée nécessaire pour développer un produit, un service ou une fonctionnalité et le/la rendre disponible sur le marché. En d'autres termes, c'est le laps de temps entre le début du développement d'un produit et son lancement sur le marché.

Bibliographie

[1] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/DXC_Technology Wikipedia Site : Présentation DXC Technology.

[2] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/DXC_Technology_Maroc Wikipédia Site : Présentation et historique DXC Technology Maroc.

[3] **[En ligne].** Disponible : https://www.rekrute.com/dxc-technology-maroc-emploi-recrutement-4995.html%23 Présentation de DXC Technology Maroc

[4] **[En ligne].** Disponible : https://www.charika.ma/societe-internationale-dxc-technology-company-4886yP Statut juridique de DXC Technology Maroc.

[5] **[En ligne].** Disponible : https://www.dxc-maroc.com/they-trust-us Principaux clients de DXC Technology Maroc.

[6] **[En ligne].** Disponible : https://dxc.com/us/en/about-us/partner-ecosystem Partenaires stratégiques de DXC Technology.

[7] **[M. BAIDOURI Hicham, Enseignant INPT].** Document du cours « Urbanisation des SI » chapitre 3 p 15.

[8] **[En ligne].** Disponible : https://www.cybermedian.com/the-20-most-frequently-mentioned-rules-and-guidelines-in-scrum/ Image Architecture de Scrum.

[9] **[En ligne].** Disponible : https://image.slidesharecdn.com/presentation1-13409802931902-phpapp02-120629094102-phpapp02/95/project-equivalent-to-two-classes-presentationprsentation-de-projet-p2m-16-728.jpg?cb=1340962955 image besoins non fonctionnels.

[10] **[En ligne].** Disponible : https://blog.coderco.io/p/monoliths-vs-microservices-a-guide Architecture Microservices

[11] **[En ligne].** Disponible : https://mindmajix.com/kafka-tutorial Architecture Kafka

[12] **[En ligne].** Disponible : https://image3.slideserve.com/5437142/uml-views1-l.jpg Différentes vues du langage UML.

[13] **[En ligne].** Disponible : https://lp.jetbrains.com/intellij-idea-features-promo/?msclkid=35511c63258d1dbaaba47a3c24f3e1db&utm_source=bing&utm_medium=cpc&utm_campaign=EMEA_en_AFRICA_IDEA_Branded&utm_term=intellij&utm_content=intellij%20idea Description et logo de l'IDE Intellij idea

[14] **[En ligne].** Disponible : https://code.visualstudio.com/Docs Description et logo de l'IDE Visual studio code.

[15] **[En ligne].** Disponible : https://www.postman.com/product/what-is-postman/ Description et logo de l'outil Postman.

[16] **[En ligne].** Disponible : https://docs.docker.com/ Description et logo de l'outil Docker.

[17] **[En ligne].** Disponible : https://en.wikipedia.org/wiki/Apache_Kafka Description et logo de l'outil Appach Kafka

[18] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/Java_(langage) Description et logo du langage Java.

[19] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/JavaScript Description et logo du langage Javascript

[20] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/TypeScript Description et logo du langage TypeScript.

[21] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/Hypertext_Markup_Language Description et logo du langage HTML.

[22] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/Sass_(langage) Description et logo du langage SCSS.

[23] **[En ligne].** Disponible : https://fr.wikipedia.org/wiki/Angular Description et logo du Framework Angular.

[24] **[En ligne].** Disponible : https://www.baeldung.com/spring-boot Description et logo du Framework Spring boot.

[25] **[En ligne].** Disponible : https://spring.io/guides/gs/accessing-data-jpat Description et logo du Framework Spring Data Jpa.

[26] **[En ligne].** Disponible : https://www.baeldung.com/security-spring Description et logo du Framework Spring security.

[27] **[En ligne].** Disponible : https://www.baeldung.com/library/spring-microservices-guide Description et logo du Framework Spring cloud.

[28] **[En ligne].** Disponible : https://spring.io/projects/spring-ai Description et logo du Framework Spring AI.

[29] **[En ligne].** Disponible : https://www.baeldung.com/mapstruct Description et logo du Framework Mapstsruct.

[30] **[En ligne].** Disponible : https://www.everything-java.com/introduction-to-lombok/ Description et logo du plugin Lombok.

[31] **[En ligne].** Disponible : https://www.baeldung.com/java-email Description et logo de la dépendance Java Mail sender.

[32] **[En ligne].** Disponible : https://www.baeldung.com/swagger-generate-pdf Description de la dépendance Swagger.

[33] **[En ligne].** Disponible : https://www.betterbuys.com/bi/reviews/tibco-jaspersoft-business-intelligence/ Description et logo de la dépendance Jasper .

[34] **[En ligne].** Disponible : https://www.baeldung.com/tracing-services-with-zipkin Description et logo de la dépendance Zipkin.

[35] **[En ligne].** Disponible : https://www.baeldung.com/spring-boot-prometheus Description et logo de la dépendance Prometheus & Grafana.

[36] **[En ligne].** Disponible : https://medium.com/commencis/what-is-websocket-43bba22fbbfe Description et logo du protocole Websockets.

[37] **[En ligne].** Disponible : https://www.baeldung.com/security-spring Description et logo du SGBDR Postgresql

[38] **[En ligne].** Disponible : https://www.baeldung.com/security-spring Description et logo de SGBD MongoDb.